Pattern Recognition

(EE5907)

Song Bai

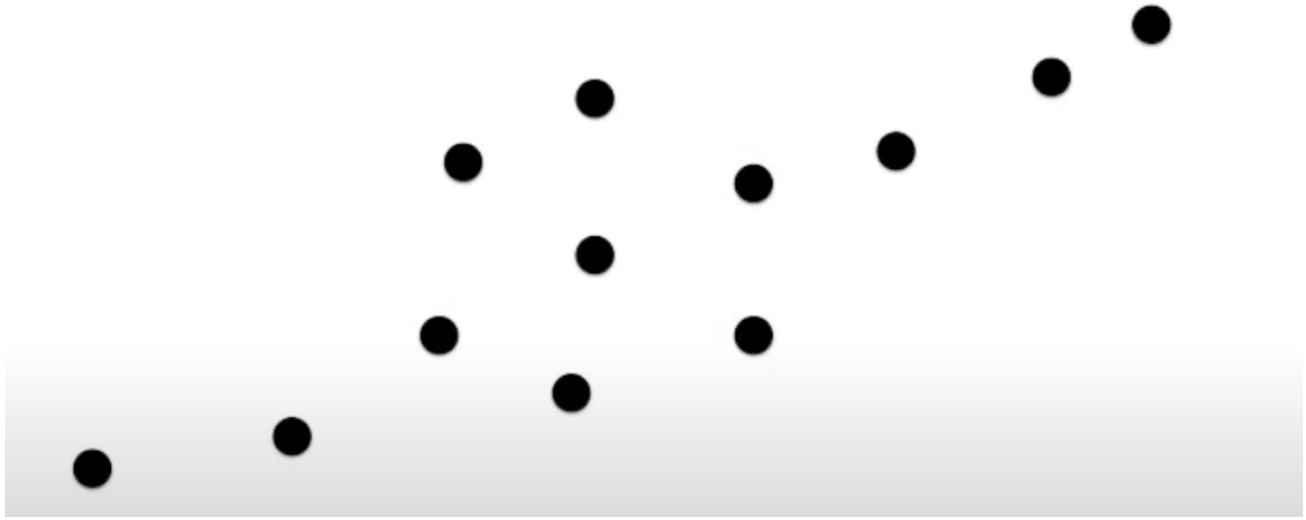
Email: songbai.site@gmail.com

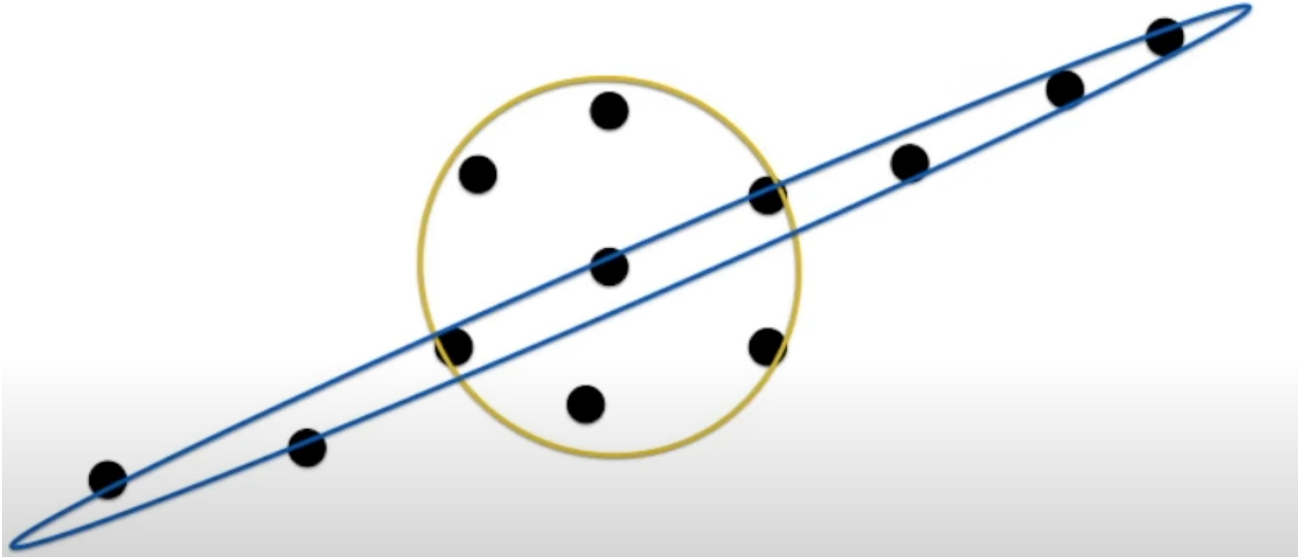
Outlines

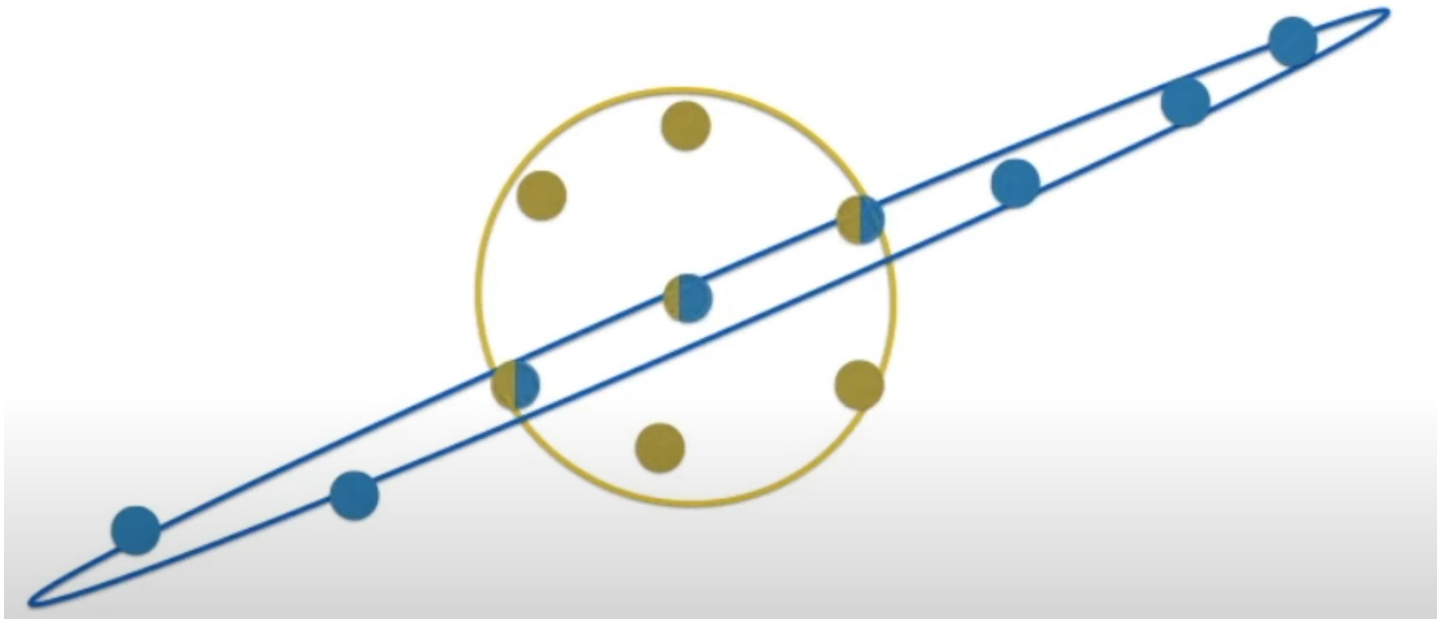
- Unsupervised Feature Extraction (PCA, NMF,…)
- Supervised Feature Extraction (LDA, GE, …)
- Clustering and Applications
- **Gaussian Mixture Model and Boosting**
- Support Vector Machine
- Deep Learning

Some slides are from Antonio Torralba

Gaussian Mixture Model (GMM)







Mixture Models

从形式上看，混合模型是一些概率密度函数（pdfs）的加权和，其中的权重由分布决定。

- Formally a Mixture Model is the weighted sum of a number of probability density functions (pdfs) where the weights are determined by a distribution, π

$$p(x) = \pi_1 f_1(x) + \pi_2 f_2(x) + \ldots + \pi_K f_K(x)$$

where $\sum_{i=1}^{K} \pi_i = 1$

$$p(x) = \sum_{i=1}^{K} \pi_i f_i(x)$$

Gaussian Mixture Models

- GMM: the weighted sum of a number of **Gaussians** where the weights are determined by a distribution, π

GMM：若干高斯的加权和，
其中的权重由分布决定。

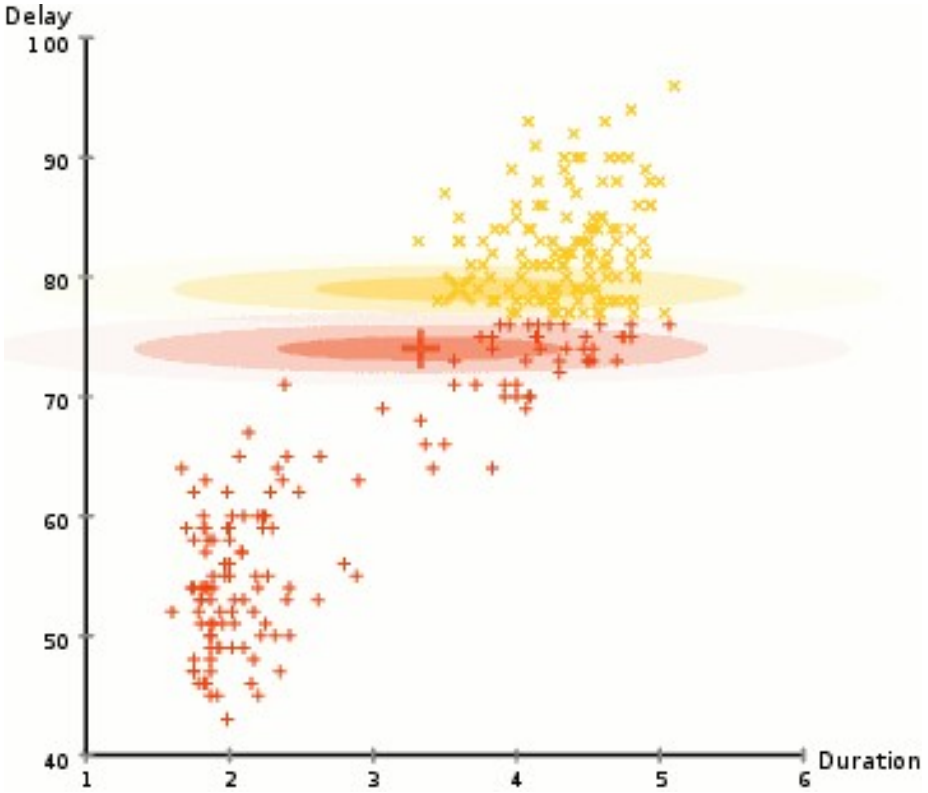
$$p(x) = \pi_1 N(x|\mu_1, \Sigma_1) + \pi_2 N(x|\mu_2, \Sigma_2) + \ldots + \pi_K N(x|\mu_K, \Sigma_K)$$

where $\sum_{i=1}^{K} \pi_i = 1$

$$p(x) = \sum_{i=1}^{K} \pi_i N(x|\mu_i, \Sigma_i)$$

Gaussian Mixture Models

- Rather than identifying clusters by “nearest” centroids
- Fit a Set of K Gaussians to the **unlabeled** data
- Maximum Likelihood over a mixture model

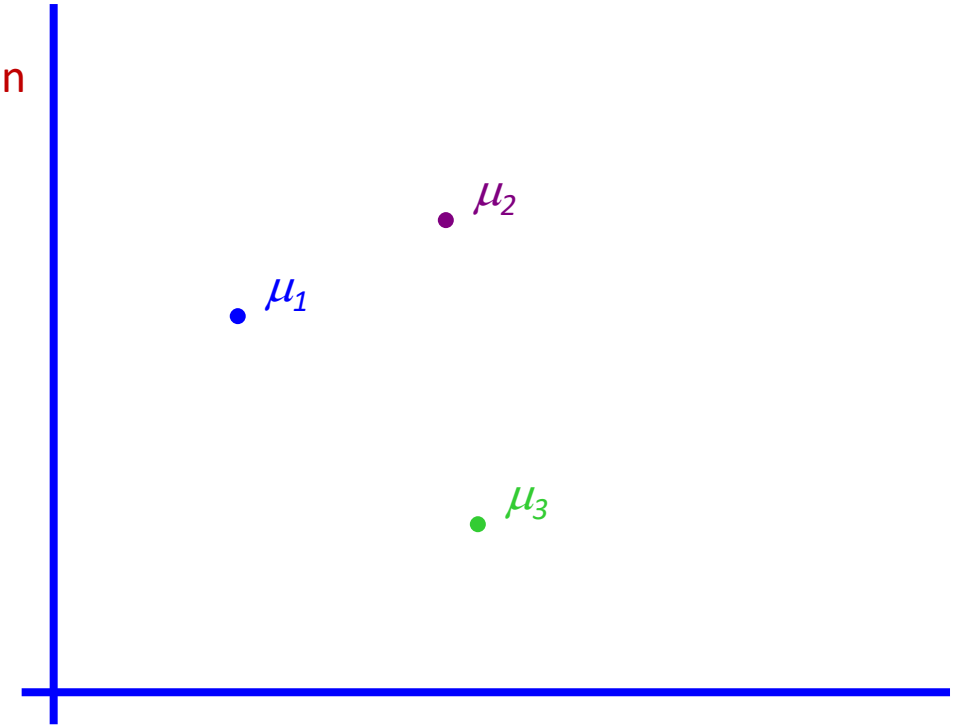


- 而不是通过 "最近的 "中心点来识别集群
- 对未标记的数据拟合一组高斯。
- 混合模型的最大似然法

The GMM Assumption

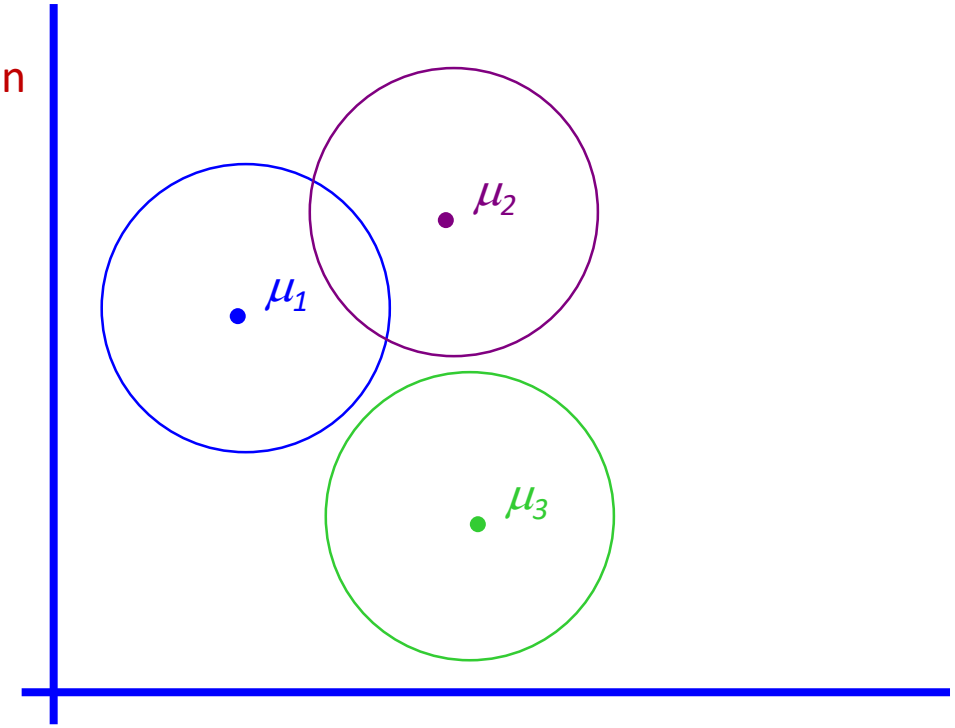
- There are K components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i

- 有一些组件。第i个分量被称为w
- 分量w有一个相关的平均向量m



The GMM Assumption

- There are K components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian model with mean μ_i and covariance matrix $\sigma^2\boldsymbol{I}$

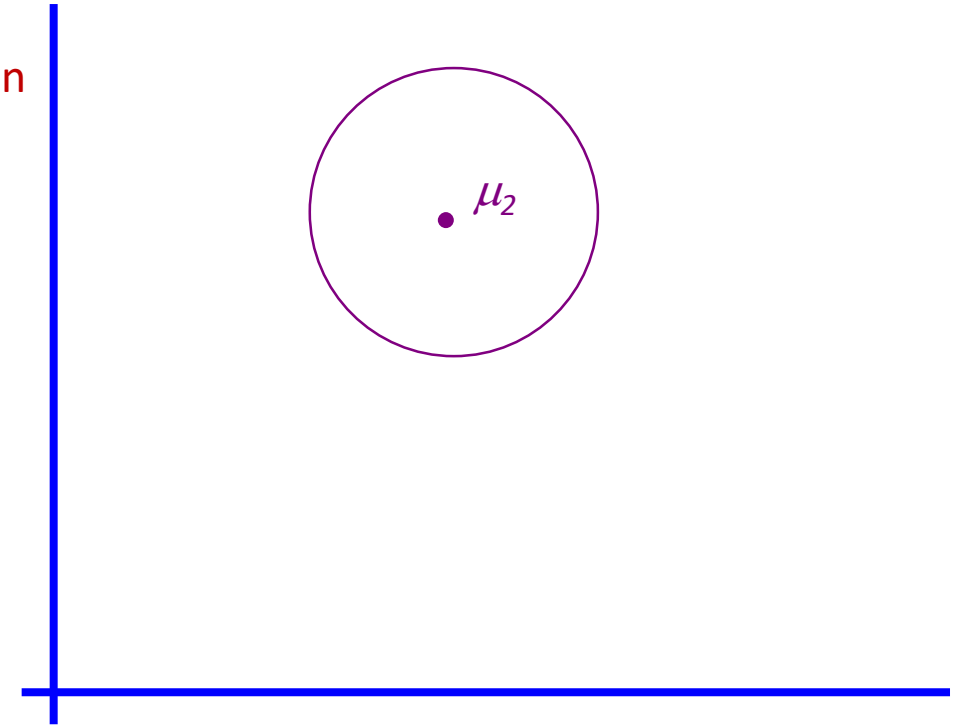


The GMM Assumption

- There are K components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian model with mean μ_i and covariance matrix $\sigma^2\boldsymbol{I}$

Assume that each data point is generated according to the following recipe:

1. Pick a component at random. Choose component *i* with probability $P(\omega_i)$.

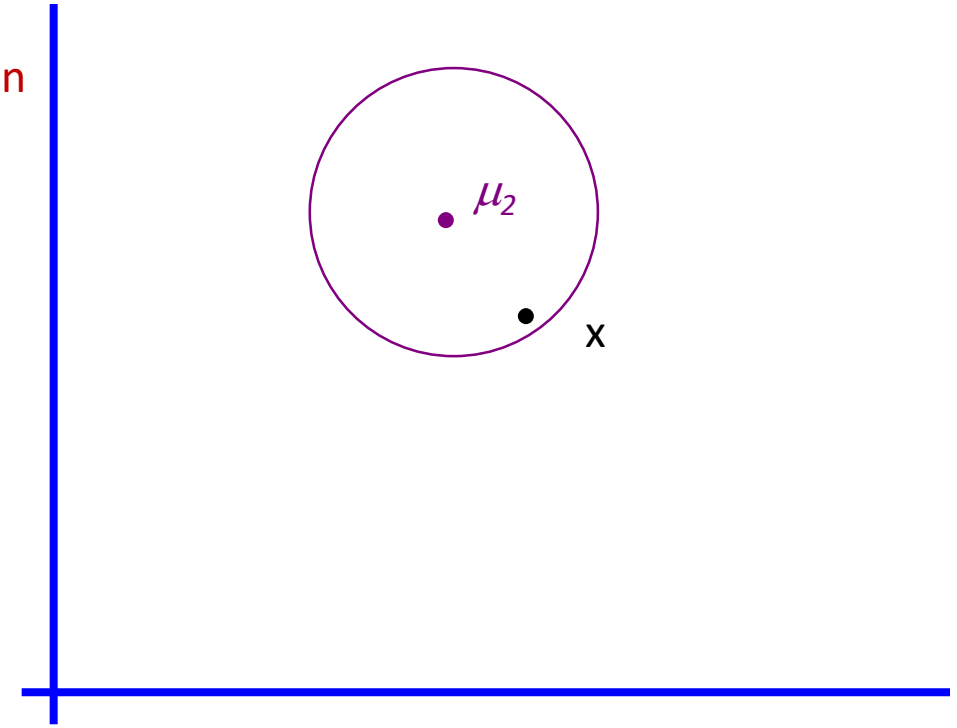


The GMM Assumption

- There are K components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian model with mean μ_i and covariance matrix $\sigma^2\boldsymbol{I}$

Assume that each data point is generated according to the following recipe:

1. Pick a component at random. Choose component *i* with probability $P(\omega_i)$.
2. Data point ~ N(μ_i, $\sigma^2\boldsymbol{I}$)



- 有一些组件。第i个分量被称为w
- 分量w有一个相关的均值向量m
- 每个分量从一个高斯模型中生成数据，其平均值为m，协方差矩阵为S

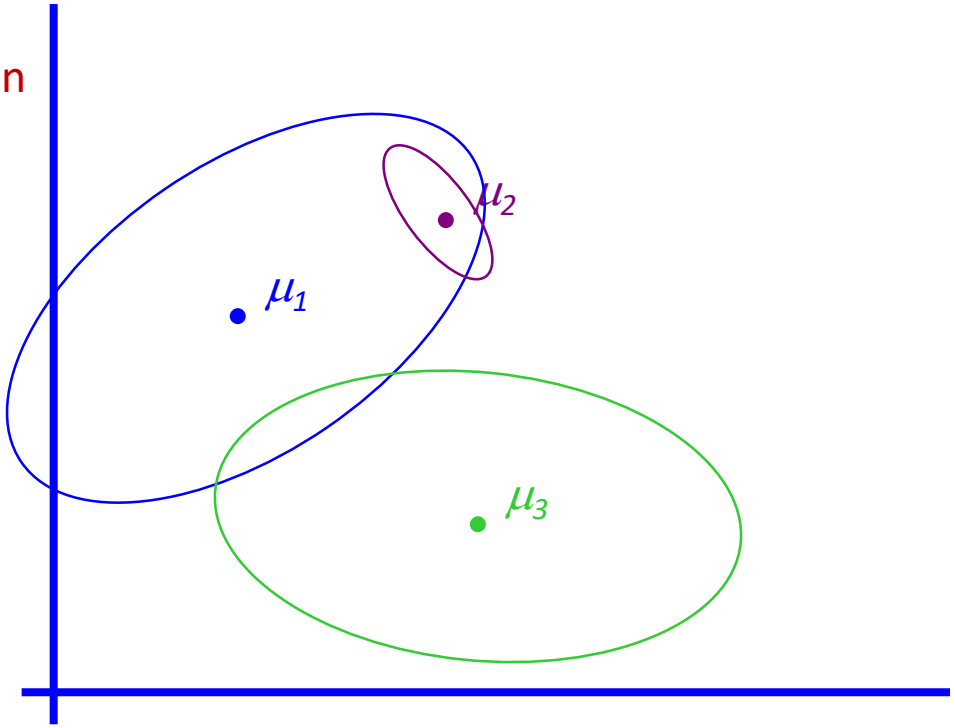
假设每个数据点都是按照以下配方生成的。

The General GMM Assumption

- There are K components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian model with mean μ_i and covariance matrix Σ_i

Assume that each data point is generated according to the following recipe:

1. Pick a component at random. Choose component *i* with probability $P(\omega_i)$.
2. Data point ~ N(μ_i, Σ_i)



1. 随机挑选一个组件。以概率P(w)选择组件i。
2. 数据点 ~ N(m , S)

The EM Algorithm

- **Expectation-maximization** (**EM**) is a method for finding maximum likelihood (or maximum a posteriori) estimate of parameter(s) in statistical model, where the model depends on **unobserved** latent variables.

预期最大化（EM）是一种在统计模型中寻找参数的最大似然（或最大后验）估计的方法，该模型取决于未观察到的潜在变量。

Latent variables are the key properties for EM.

The EM Algorithm

- **EM** is an **iterative** method which alternates between performing an **E**xpectation (**E**) step and a **M**aximization (**M**) step
  - **E-step** computes the expectation of the log-likelihood evaluated using the current estimated distributions for the latent variables based on the parameters inferred from previous step
  - **M-step** computes parameters maximizing the expected log-likelihood from the E-step. These parameter-estimates are then used to determine the distribution of the latent variables in the next E-step.

Latent variables become constants here.

- EM是一种迭代方法，在期望（E）步骤和最大化（M）步骤之间交替进行。
- E步计算对数可能性的期望值，该期望值是根据前一步推断出的参数，使用潜变量的当前估计分布来评估的。
- M-步骤计算参数，使E-步骤中的预期对数似然最大化。然后，这些参数估计值被用来确定下一个E-步骤中潜变量的分布。

Compute Likelihood

- We define:

$$\pi_i = P(\omega_i) \quad \text{where} \quad \sum_i \pi_i = 1$$

$$z_i = p(\omega_i|x) = \frac{P(\omega_i)p(x|\omega_i)}{\sum_{j=1}^{K} P(\omega_j)p(x|\omega_j)}$$

$$z_k^n = p(\omega_k|x_n)$$

- Identify a likelihood function

$$p(x_1, \ldots, x_N|\pi, \mu) = \prod_{n=1}^{N} p(x_n|\pi, \mu)$$

x_n's were drawn independently

$$= \prod_{n=1}^{N} \sum_{k=1}^{K} p(x_n|\omega_k, \mu_k)P(\omega_k)$$

Maximum Likelihood over a GMM

- Identify a log-likelihood function

$$\ln p(x_1, \ldots, x_n|\pi, \mu) = \sum_{n=1}^{N} \ln \left[\sum_{k=1}^{K} p(x_n|\omega_k, \mu_k) P(\omega_k) \right]$$

- Compute and set partials to 0

$$\frac{\partial \ln p(x_1, \ldots, x_n|\pi, \mu)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{p(x_n|\pi, \mu)} \frac{\partial \sum_{k=1}^{K} N(x_n|\mu_k) P(\omega_k)}{\partial \mu_k}$$

$$= \sum_{n=1}^{N} \frac{P(\omega_k)}{p(x_n|\pi, \mu)} \frac{\partial N(x_n|\mu_k)}{\partial \mu_k}$$

$$= \sum_{n=1}^{N} \frac{P(\omega_k) N(x_n|\mu_k)}{p(x_n|\pi, \mu)} \frac{\partial \ln N(x_n|\mu_k)}{\partial \mu_k}$$

$$p(x_n|\pi, \mu) = \sum_{k=1}^{K} p(x_n|\omega_k, \mu_k) P(\omega_k)$$

$$\frac{\partial \ln f(x)}{\partial x} = \frac{1}{f(x)} \frac{\partial f(x}{\partial x}$$

see next slide

Maximum Likelihood over a GMM

$$\frac{\partial \ln p(x_1, \ldots, x_n|\pi, \mu)}{\partial \mu_k} = \sum_{n=1}^{N} p(\omega_k|x_n) \frac{\partial \ln \exp\left(-\frac{1}{2\sigma^2}(x_n - \mu_k)^2\right)}{\partial \mu_k}$$

$$= \sum_{n=1}^{N} z_k^n \left(-\frac{1}{2\sigma^2} \frac{\partial (x_n - \mu_k)^2}{\partial \mu_k}\right)$$

$$= \sum_{n=1}^{N} z_k^n \frac{x_n - \mu_k}{\sigma^2} = 0$$

set partials to 0

$$\mu_k = \frac{\sum_{n=1}^{N} z_k^n x_n}{\sum_{n=1}^{N} z_k^n}$$

EM for General GMMs

We don't know $P(\omega_1), P(\omega_2), \ldots, P(\omega_K), \mu_1, \mu_2, \ldots, \mu_K, \Sigma_1, \Sigma_2, \ldots, \Sigma_K$

Similarly, after compute the log likelihood and take partials to 0, we have

$$\mu_k = \frac{\sum_{n=1}^{N} z_k^n x_n}{\sum_{n=1}^{N} z_k^n}$$

$$\Sigma_k = \frac{\sum_{n=1}^{N} z_k^n (x_n - \mu_k)(x_n - \mu_k)^T}{\sum_{n=1}^{N} z_k^n}$$

$$\pi_k = \frac{\sum_{n=1}^{N} z_k^n}{N}$$

Summary: EM for GMMs

- Initialize the parameters
  - Evaluate the log likelihood

- Expectation-step: Compute the expectation
- Maximization-step: Re-estimate Parameters
  - Evaluate the log likelihood
  - Check for convergence

- 初始化参数
- 评估对数似然
- 期望步骤。计算期望值
- 最大化步骤。重新估计参数
- 评估对数似然
- 检查收敛性

EM for GMMs

- E-step: Compute "expected" classes of all data points for each class

$$z_k^n = \frac{\pi_k N(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_n|\mu_j, \Sigma_j)}$$

where $\pi_k = p(\omega_k)$

EM for GMMs

- M-Step: Re-estimate Parameters

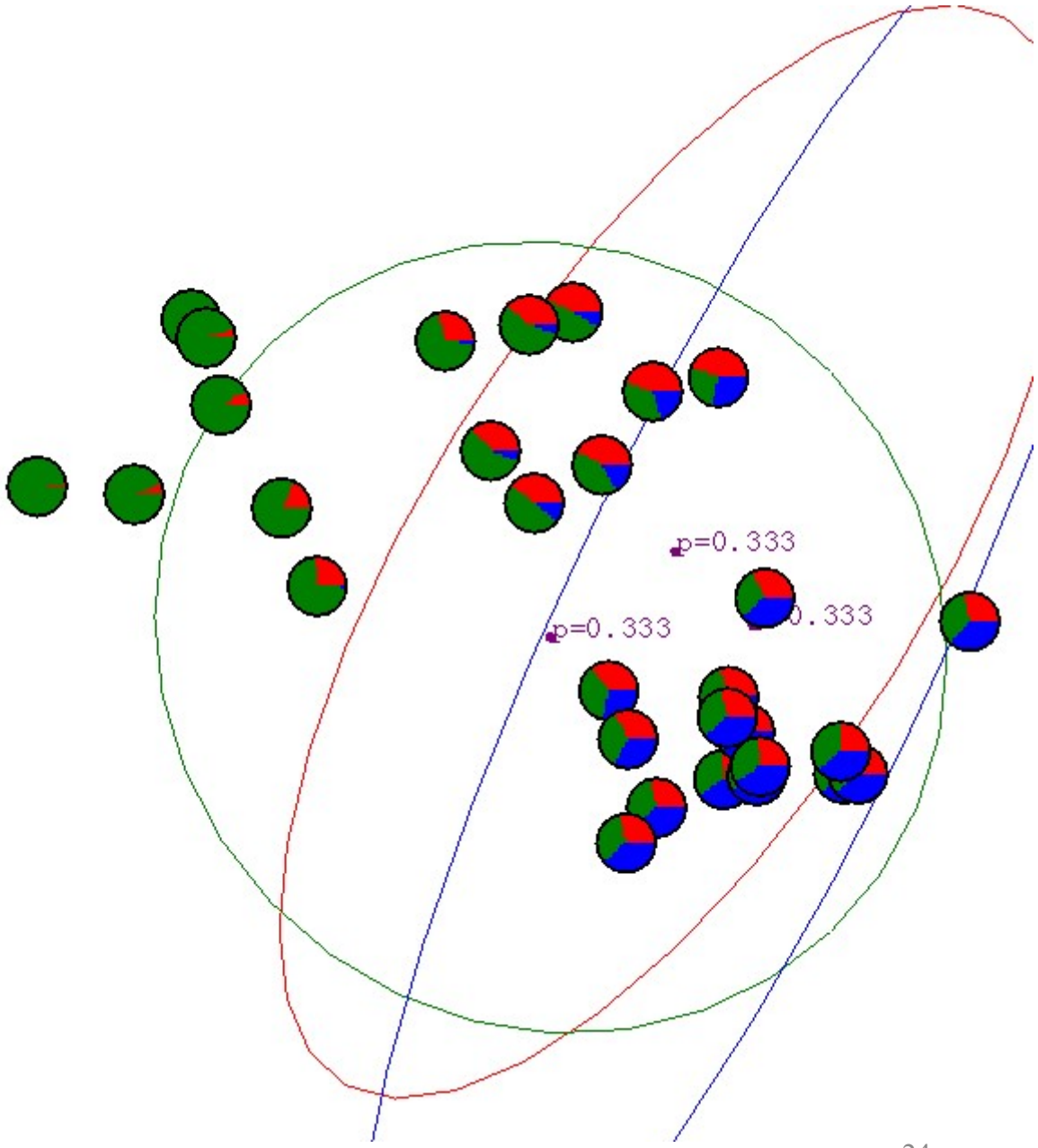
$$\mu_k^{new} = \frac{\sum_{n=1}^{N} z_k^n x_n}{\sum_{n=1}^{N} z_k^n}$$

$$\Sigma_k^{new} = \frac{\sum_{n=1}^{N} z_k^n (x_n - \mu_k^{new})(x_n - \mu_k^{new})^T}{\sum_{n=1}^{N} z_k^n}$$

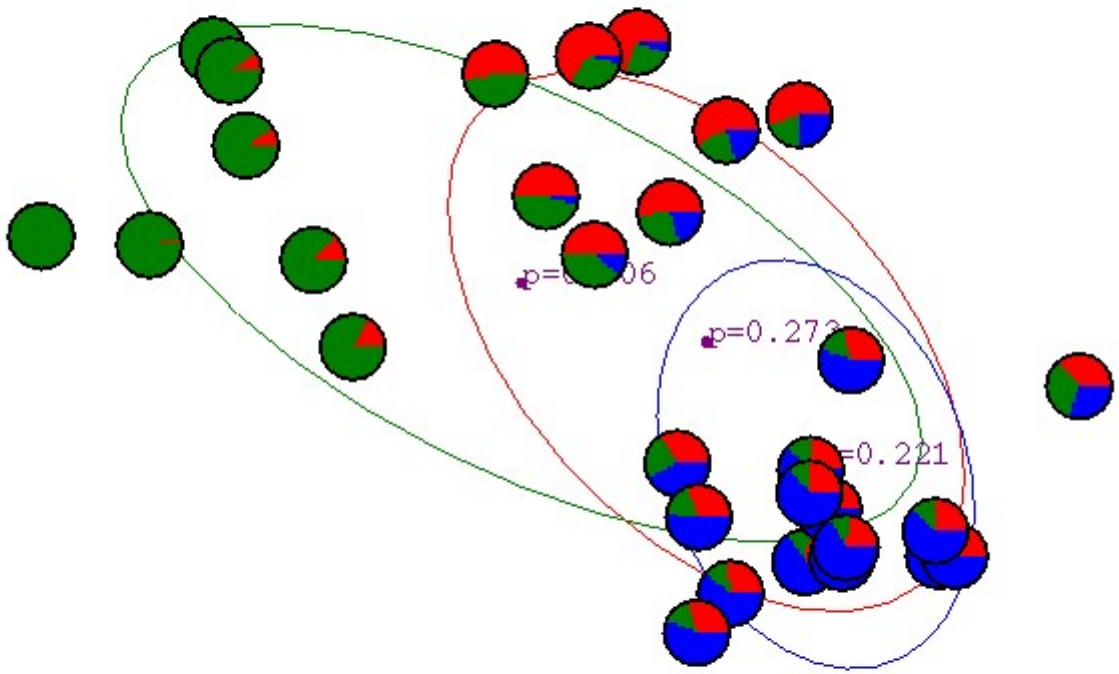
$$\pi_k^{new} = p(\omega_k)^{new} = \frac{\sum_{n=1}^{N} z_k^n}{N}$$

Latent variables become constants here.

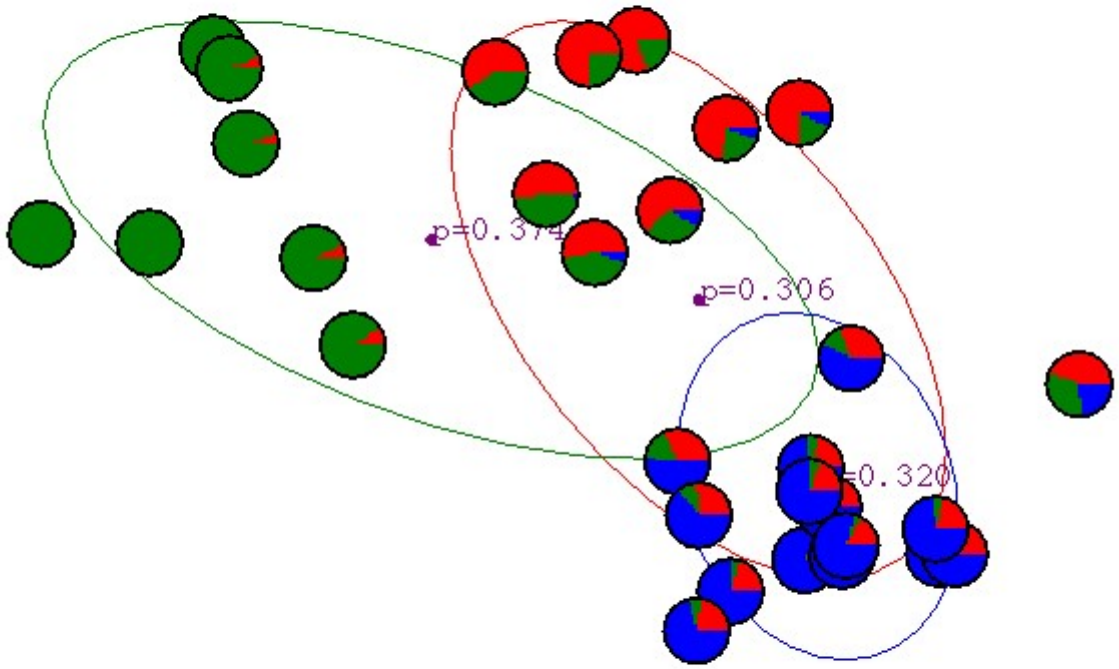
Gaussian Mixture Model Example: Start



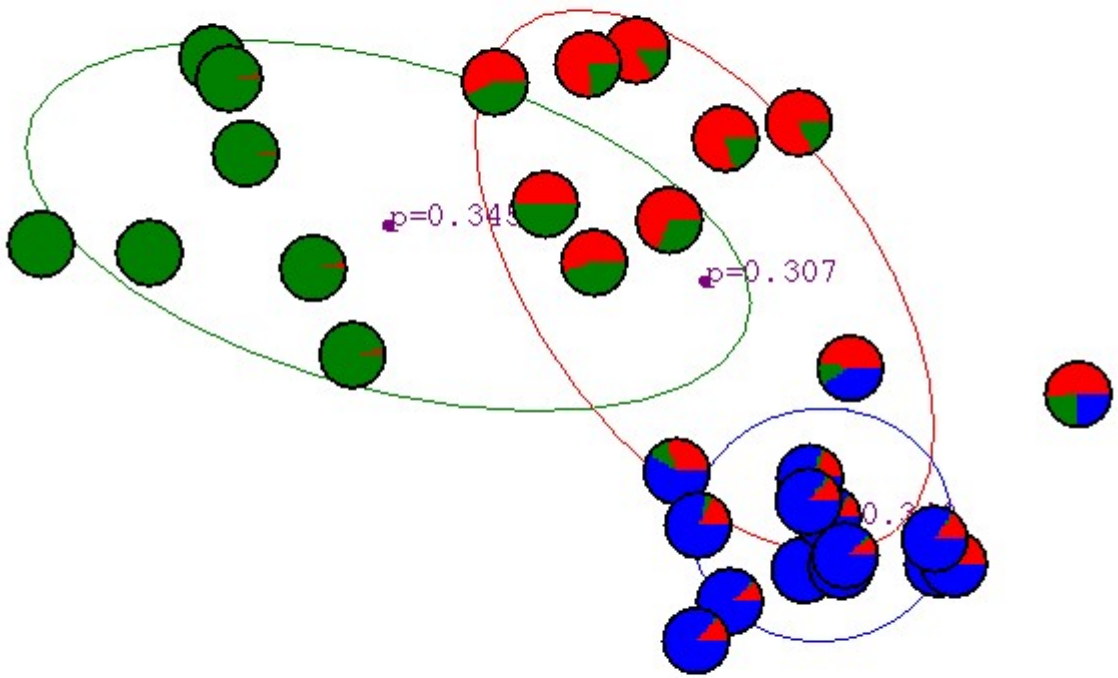
After 1st iteration



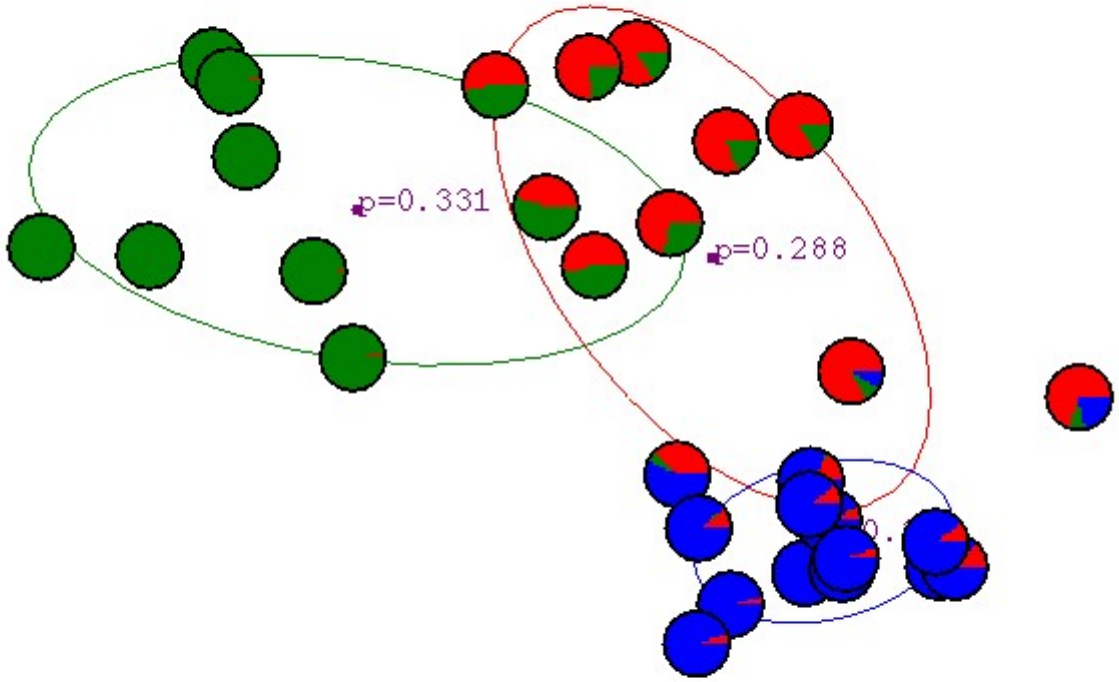
After 2nd iteration



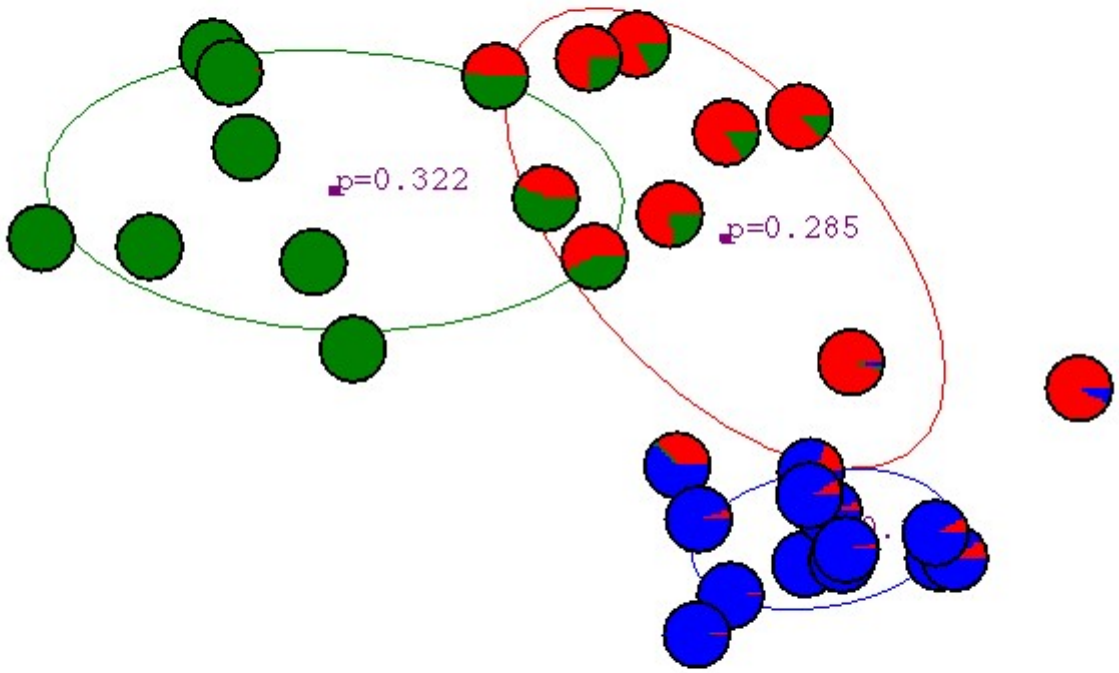
After 3rd iteration



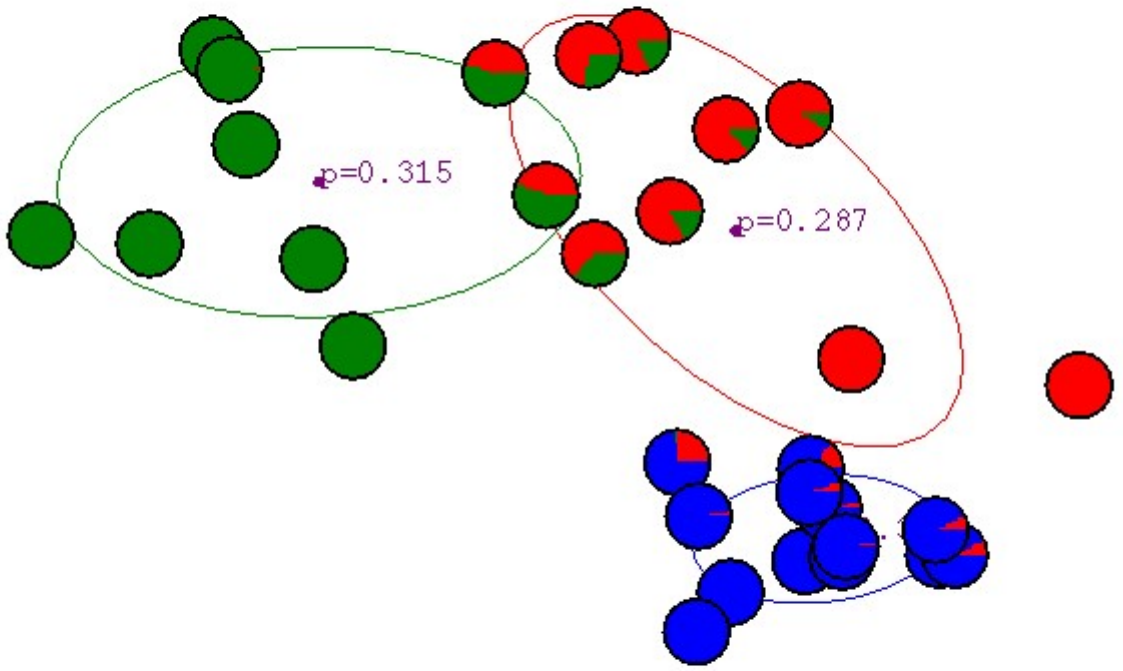
After 4th iteration



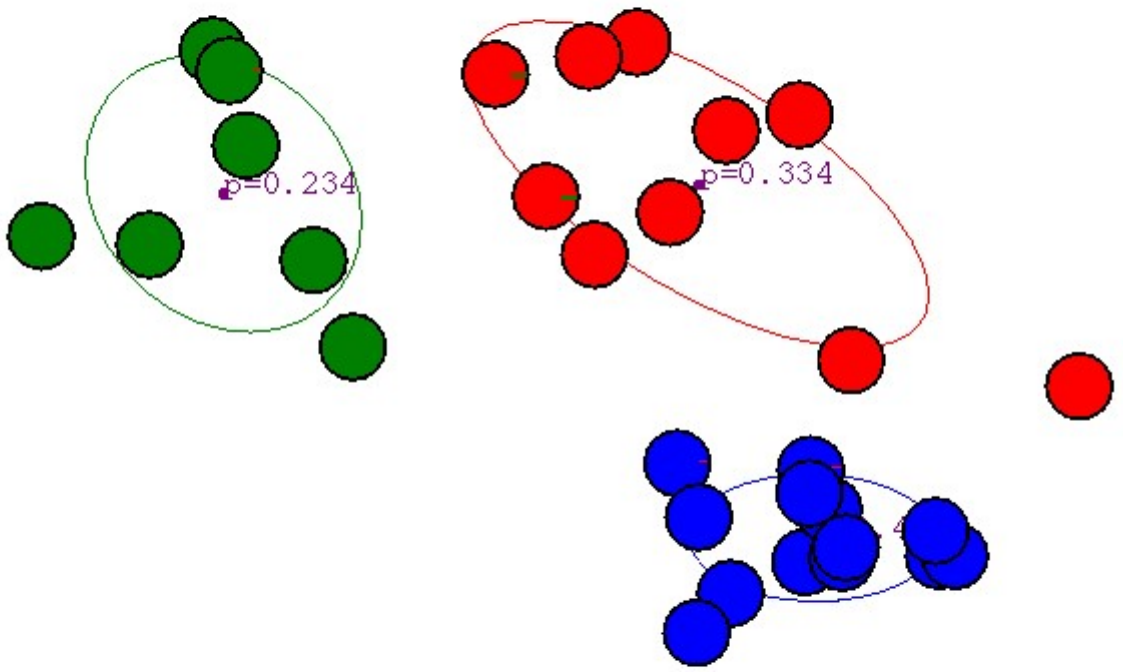
After 5th iteration



After 6th iteration



After 20th iteration

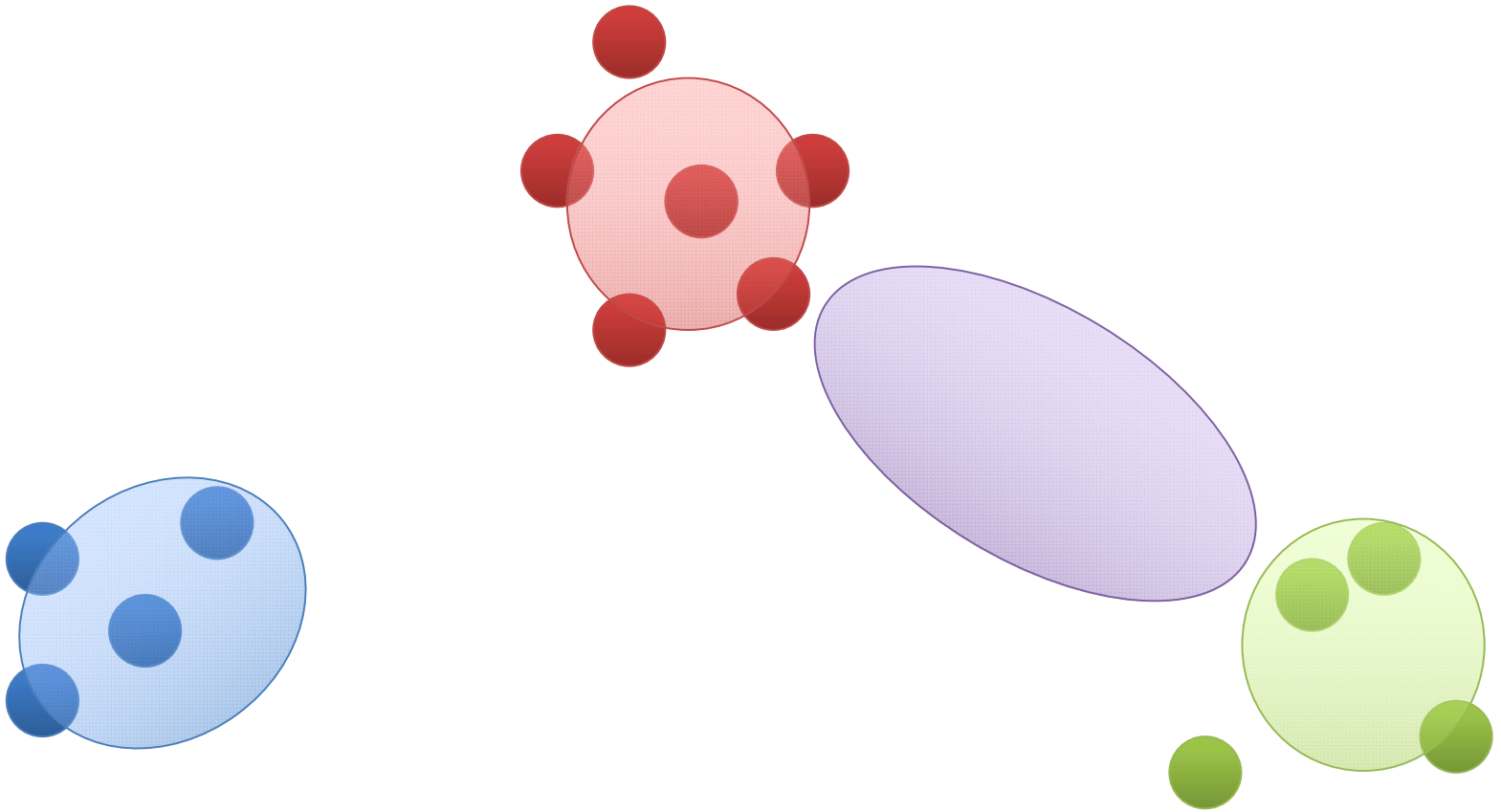


Relationship to K-means

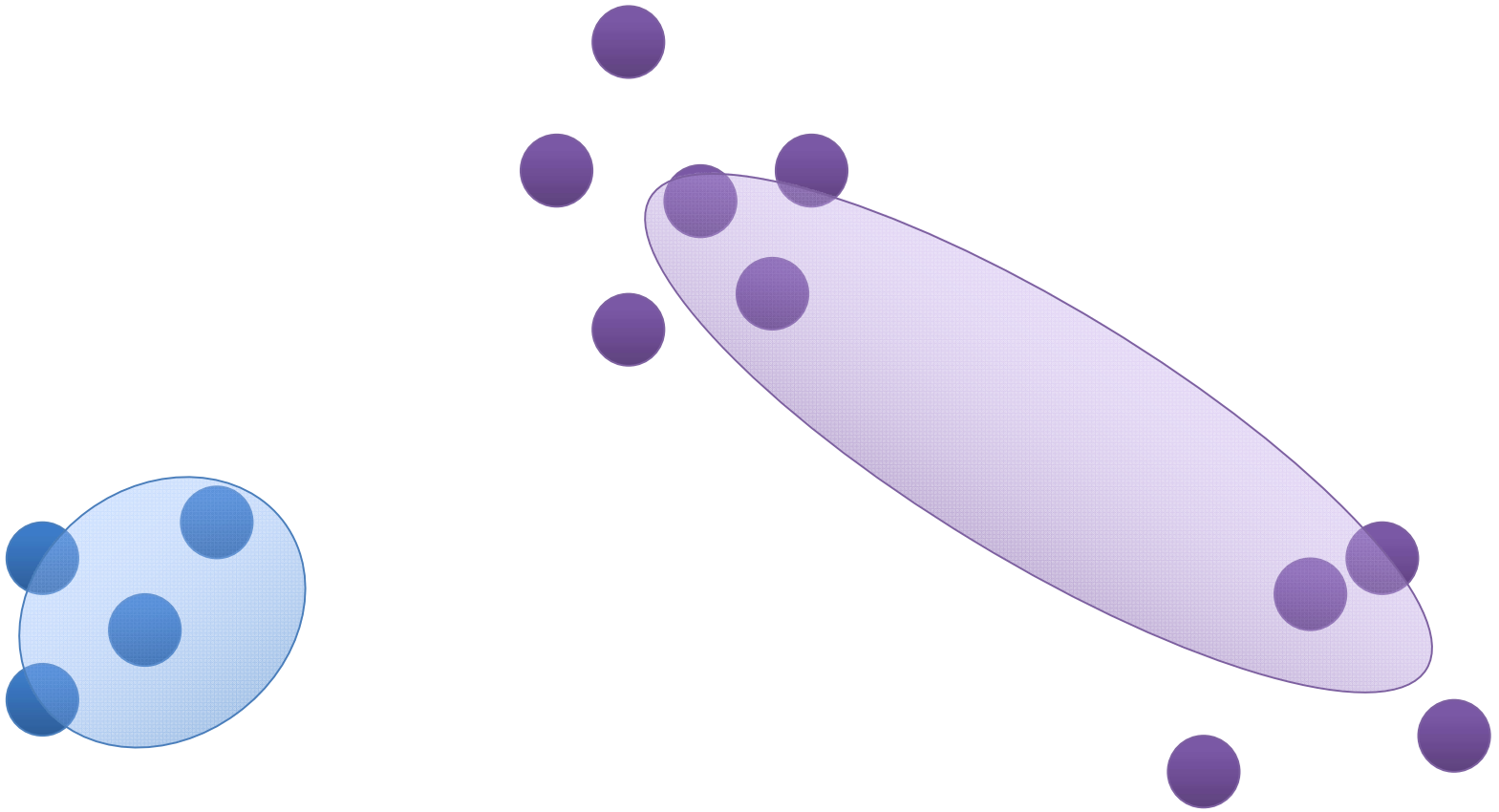
- K-means makes **hard** decisions.
  - Each data point gets assigned to a single cluster.
- GMM makes **soft** decisions.
  - Each data point yields a posterior
- Potential problem:
  - Incorrect number of Mixture Components

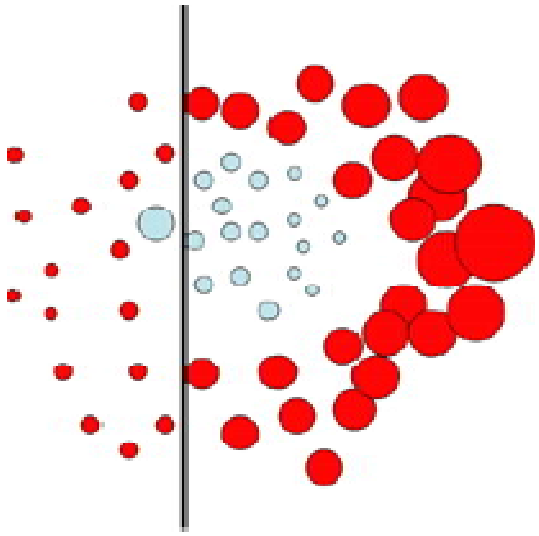
- K-means做出了强的决定。
- 每个数据点都被分配到一个群组。
- GMM做软决策。
- 每个数据点都会产生一个后验
- 潜在的问题。
- 混合成分的数量不正确

Incorrect Number of Gaussians



Incorrect Number of Gaussians





Discriminative: Boosting

Example Task

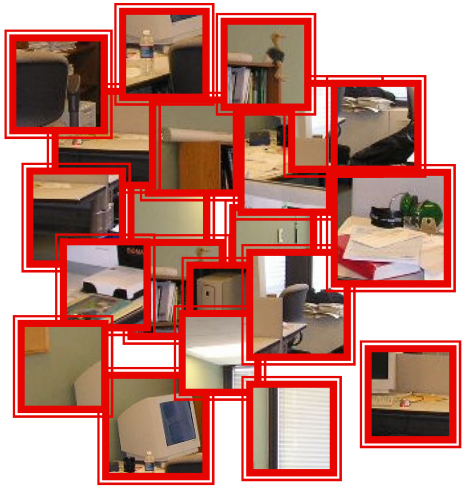
物体检测和识别被表述为一个分类问题。
图像被划分为一组重叠的窗口…并在每个窗口决定它是否包含一个目标物体。

Object detection and recognition is formulated as a classification problem.

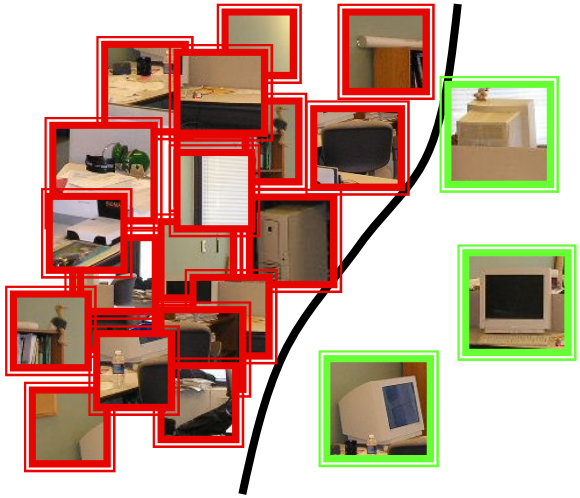
The image is partitioned into a set of overlapping windows

… and a decision is taken at each window about if it contains a target object or not.

Where are the screens?



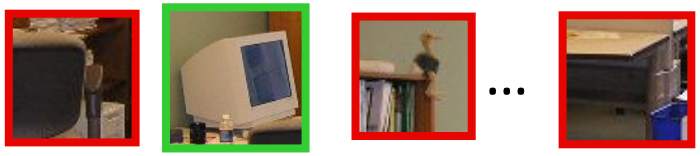
Bag of image patches



In some feature space

Formulation

- Formulation: binary classification



Features x = x_1 x_2 x_3 … x_N x_{N+1} x_{N+2} … x_{N+M}

Labels y = -1 +1 -1 -1 ? ? ?

Training data: each image patch is labeled as containing the object or background

Test data

- Classification function

$\hat{y} = F(x)$ where $F(x)$ belongs to some family of functions

- Minimize misclassification error

Why Boosting?

- A simple algorithm for learning robust classifiers
  - Freund & Shapire, 1995
  - Friedman, Hastie, Tibshhirani, 1998

- Provides efficient algorithm for sparse visual feature selection
  - *Tieu & Viola, 2000*
  - *Viola & Jones, 2003*

- Easy to implement, not requires external optimization tools

Boosting

- Defines a classifier using an additive model:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \ldots$$

Strong classifier

Features vector

Weight

Weak classifier

Boosting

定义了一个使用加性模型的分类器。

- Defines a classifier using an additive model:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \ldots$$

Strong classifier

Features vector

Weight

Weak classifier

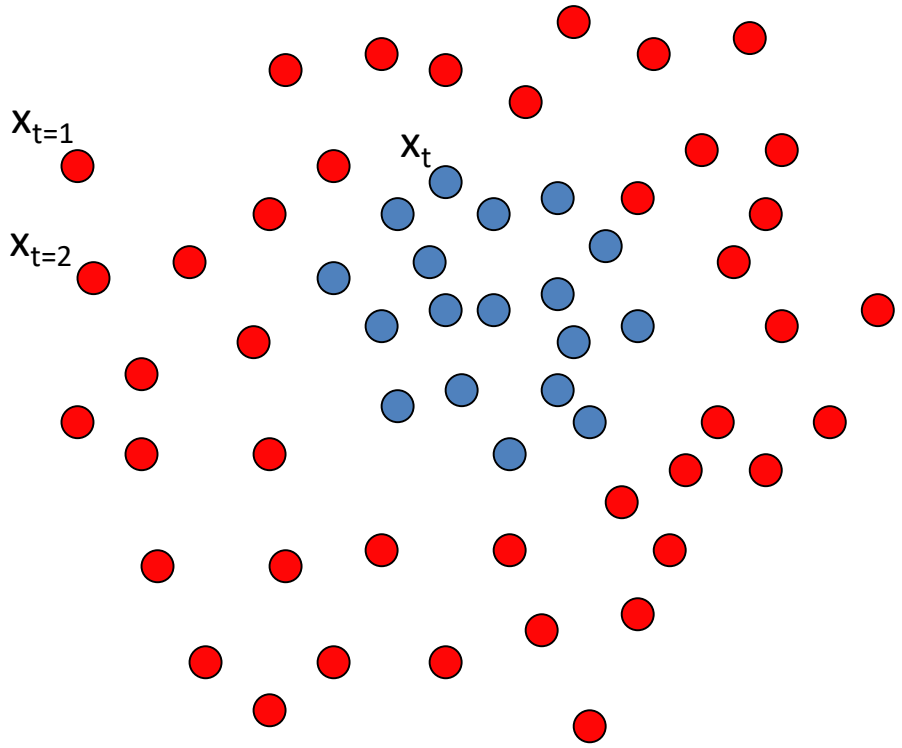
我们需要定义一个弱分类器系列

- We need to define a family of weak classifiers

$f_k(x)$ from a family of weak classifiers

Toy Example

- It is a sequential procedure:



Each data point has a class label:

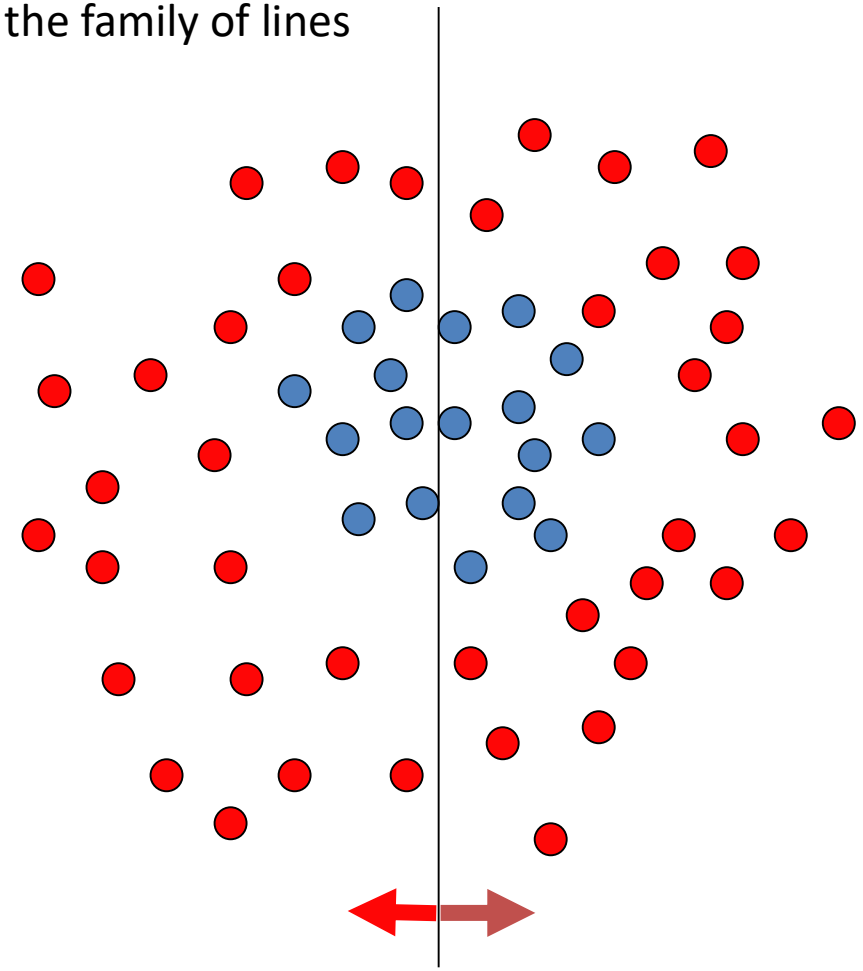
$$y_t = \begin{cases} +1 & (\bigcirc) \\ -1 & (\bigcirc) \end{cases}$$

and a weight:

$$w_t = 1$$

Toy Example

Weak learners from the family of lines



Each data point has

a class label:

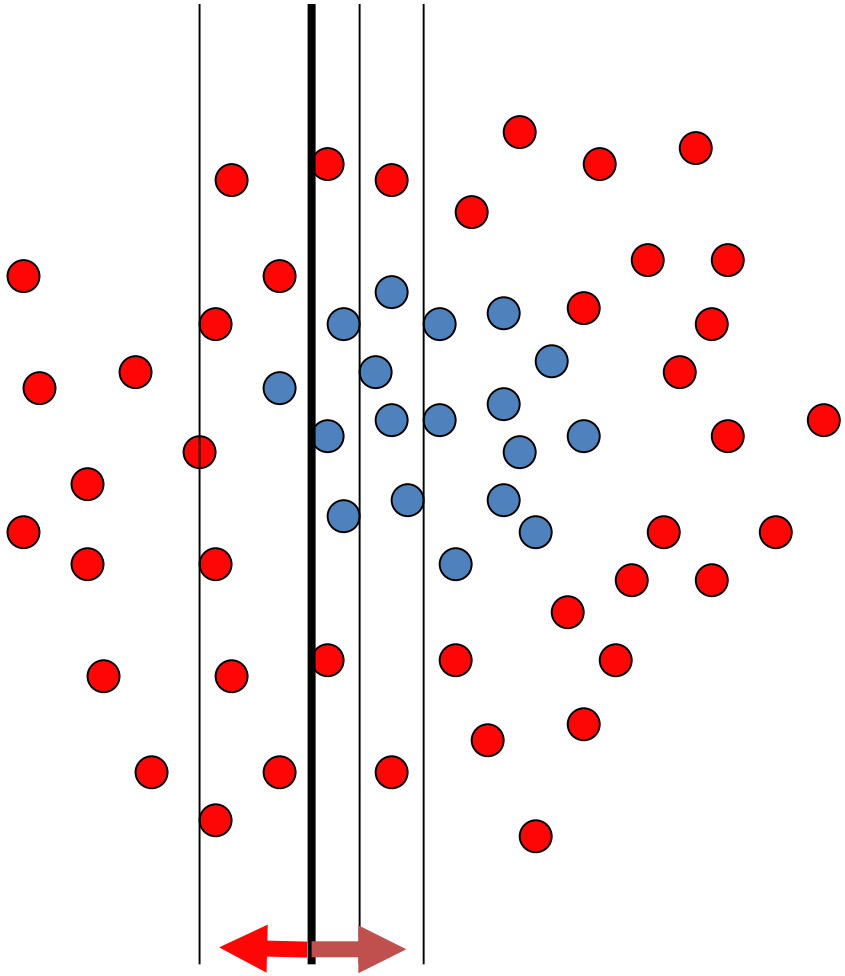
$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\bullet) \end{cases}$$

and a weight:

$$w_t = 1$$

Weak classifier h => p(error) = 0.5 it is at chance

Toy Example



Each data point has

a class label:

$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\bullet) \end{cases}$$

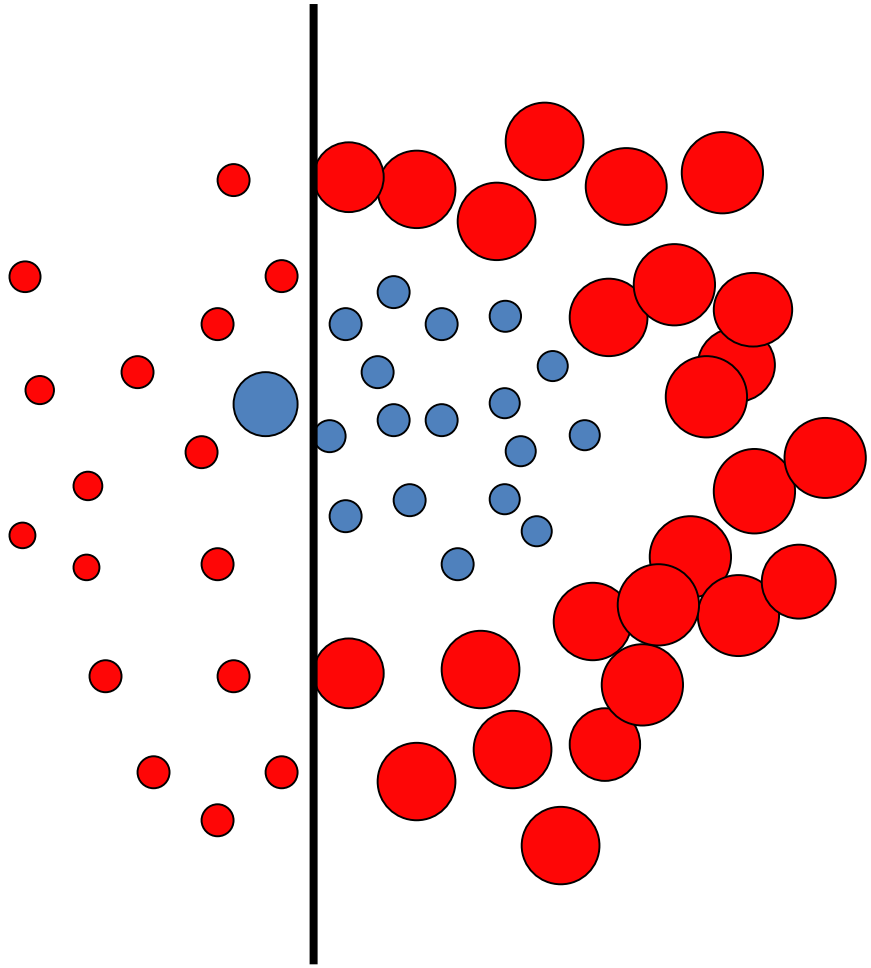
and a weight:

$$w_t = 1$$

This one seems to be the best

This is a '**weak classifier**': It performs slightly better than chance.

Toy Example



Each data point has

a class label:

$$y_t = \begin{cases} +1 & (\bigcirc) \\ -1 & (\bigcirc) \end{cases}$$

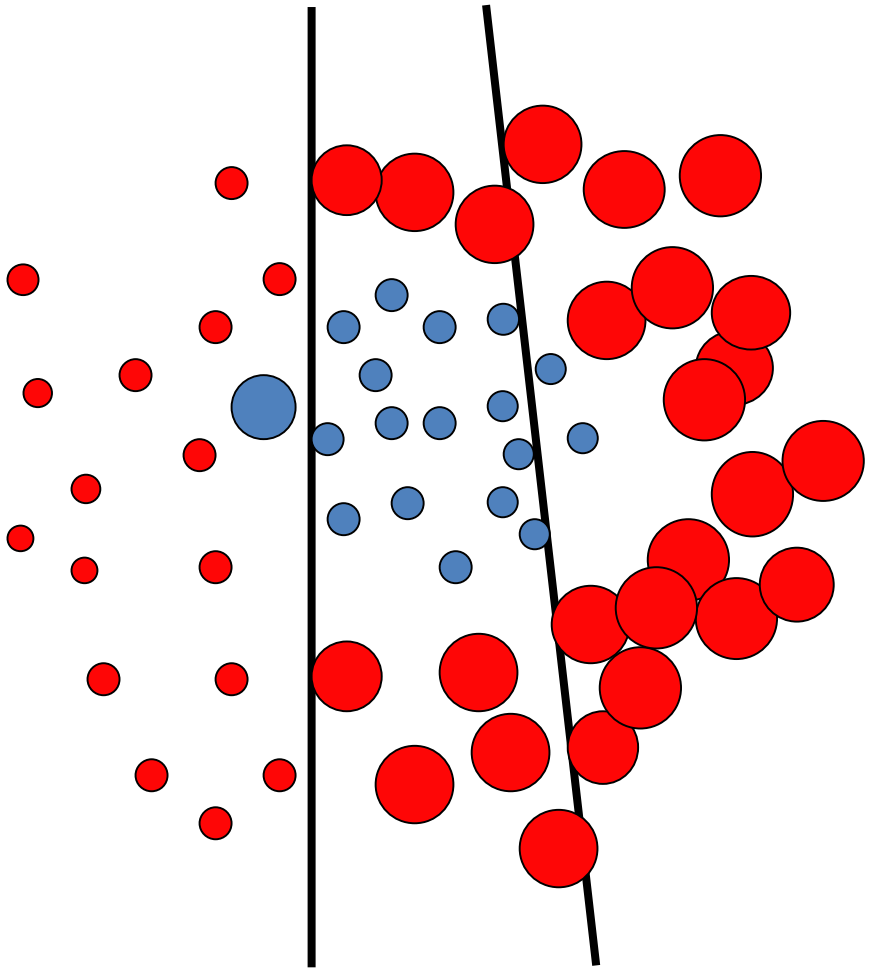
We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t F(x_t)\}$$

Current weak classifier

We set a new problem for which the current classifier performs at chance again

Toy Example



Each data point has

a class label:

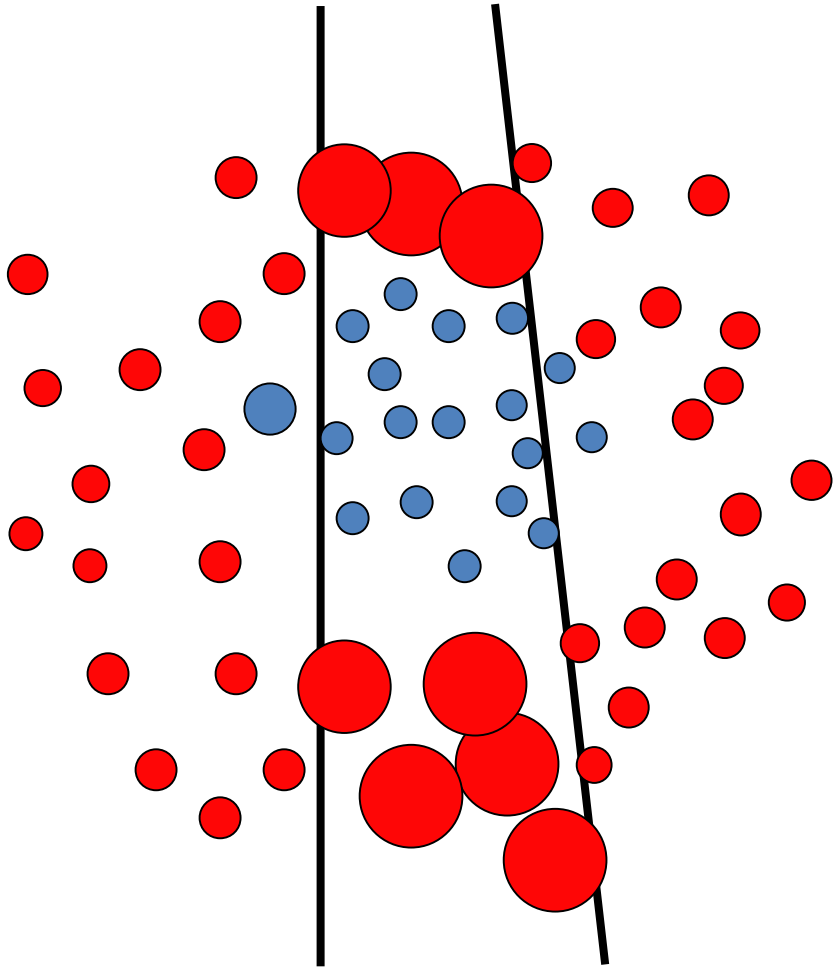
$$y_t = \begin{cases} +1 & (\bigcirc) \\ -1 & (\bigcirc) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t F(x_t)\}$$

Similarly, we learn another weak classifier

Toy Example



Reweighting

Each data point has

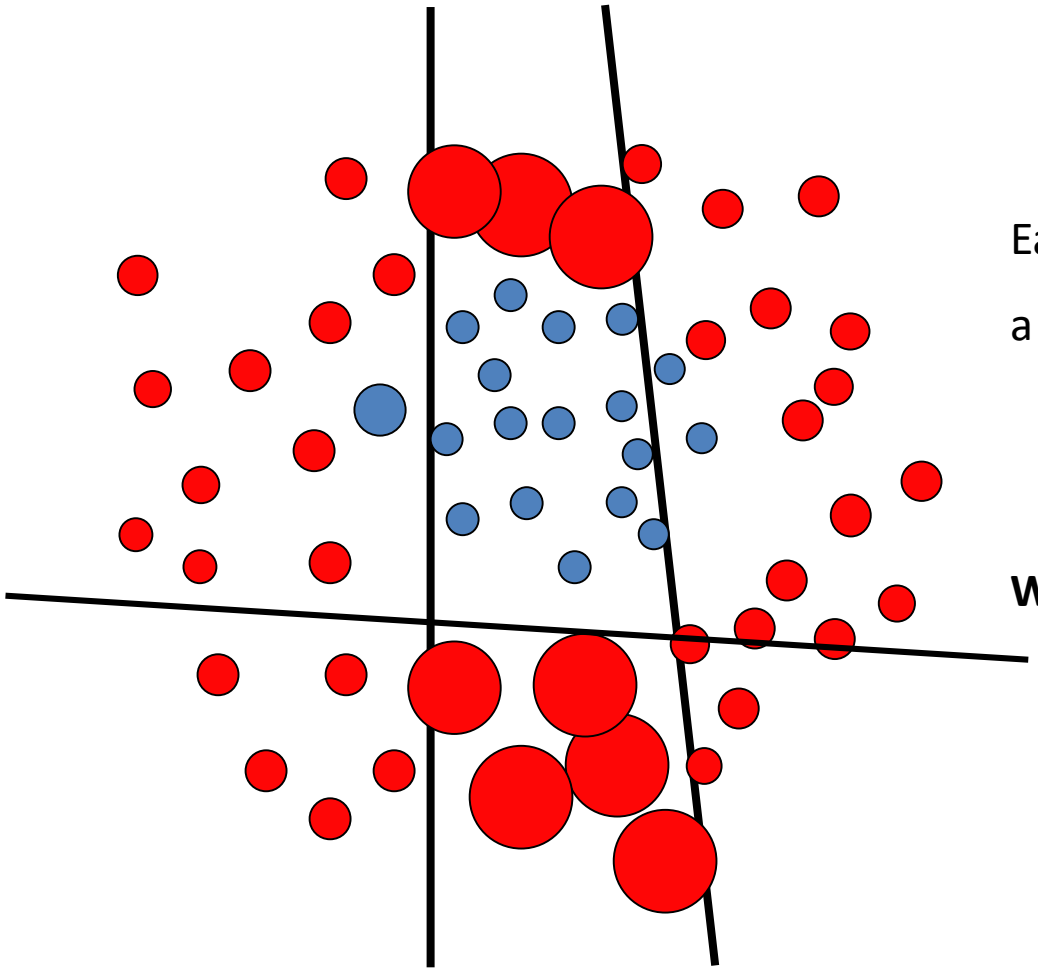
a class label:

$$y_t = \begin{cases} +1 & (\bigcirc) \\ -1 & (\bigcirc) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t F(x_t)\}$$

Toy Example



Each data point has

a class label:

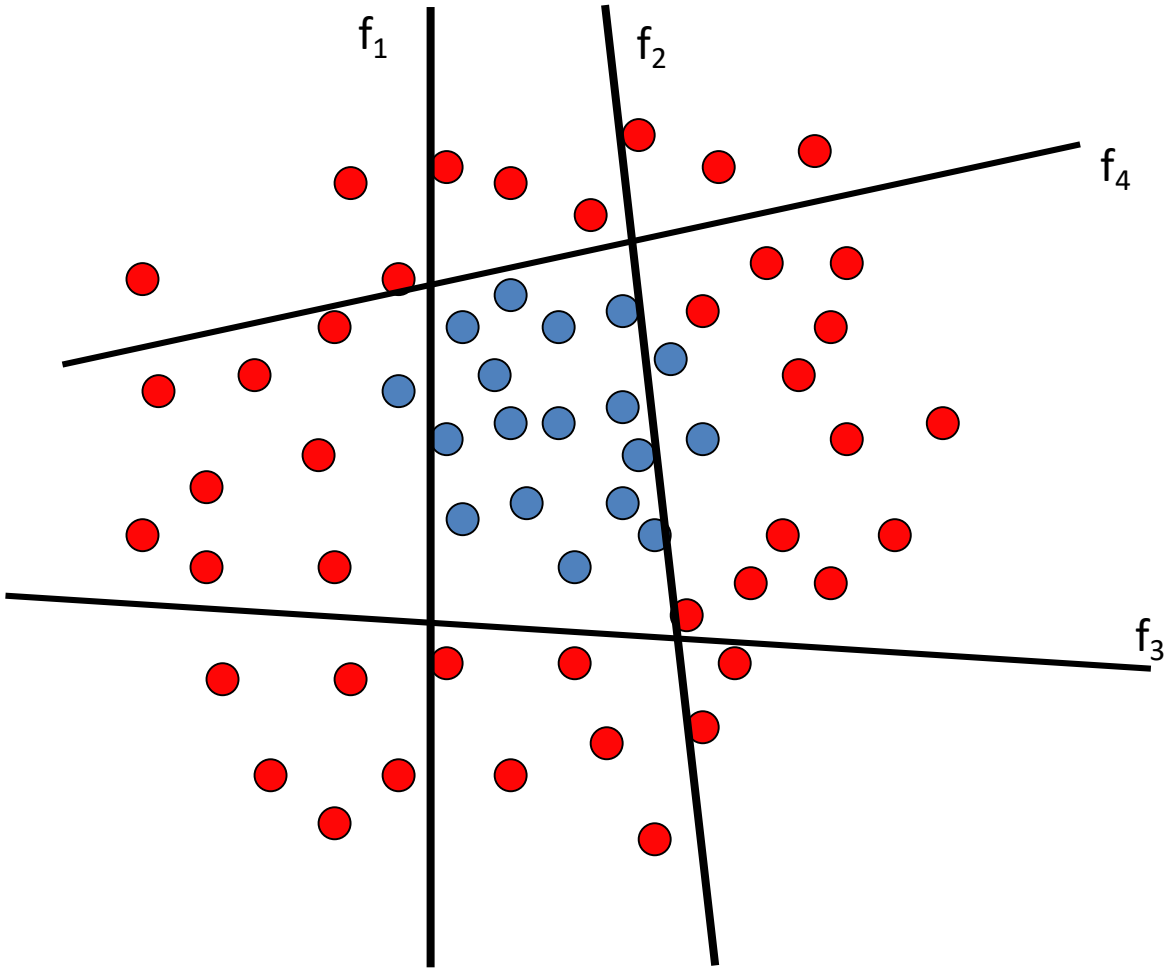
$$y_t = \begin{cases} +1 & (\bigcirc) \\ -1 & (\bigcirc) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t F(x_t)\}$$

Similarly, we learn another weak classifier

Toy Example



The strong (non- linear) classifier is built as the combination of all the weak (linear) classifiers.

Boosting

- For different cost function and minimization algorithm, the result is a different flavor of Boosting

- We shall introduce gentleBoosting
  - It is simple to implement and numerically stable.

- 对于不同的成本函数和最小化算法，其结果是不同风味的Boosting。
- 我们将介绍gentleBoosting
- 它的实现很简单，而且在数值上很稳定。

Boosting

Boosting fits the additive model

$$F(x) = f_1(x) + f_2(x) + f_3(x) + \ldots$$

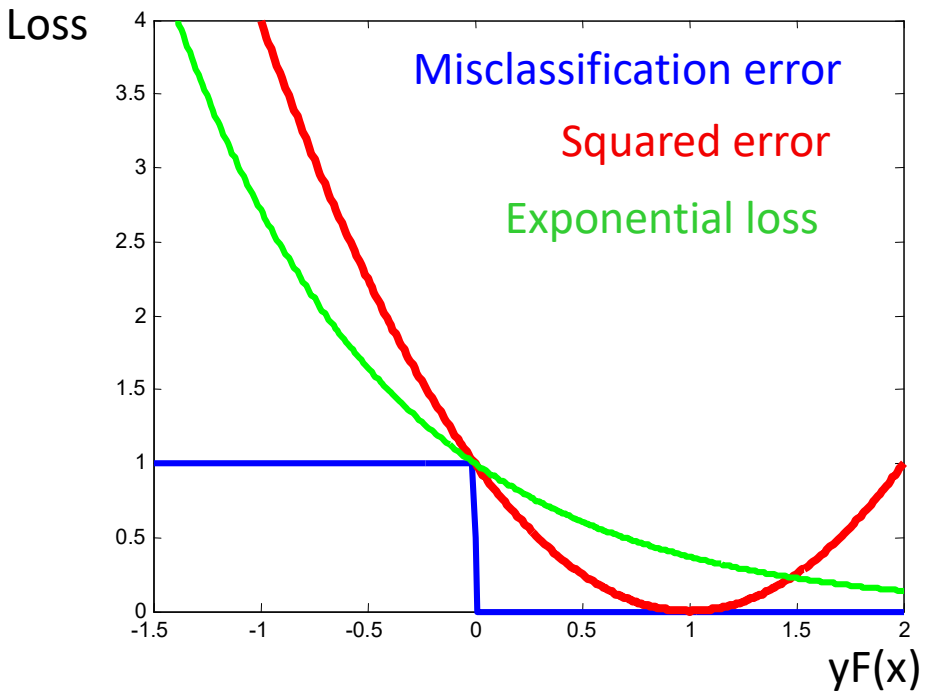
by minimizing the exponential loss

$$J(F) = \sum_{t=1}^{N} e^{-y_t F(x_t)}$$

Training samples

The exponential loss is a differentiable upper bound to the misclassification error.

Exponential Loss



Squared error

$$J = \sum_{t=1}^{N} [y_t - F(x_t)]^2$$

Exponential loss

$$J = \sum_{t=1}^{N} e^{-y_t F(x_t)}$$

Boosting

Sequential procedure. At each step m we add

$$F(x) \leftarrow F(x) + f_m(x)$$

to minimize the residual loss

$$(\phi_m) = \arg\min_{\phi} \sum_{t=1}^{N} J\left(y_t, F(x_t) + f(x_t; \phi)\right)$$

Parameters of the weak classifier

Desired output

input

For more details: Friedman, Hastie, Tibshirani. “Additive Logistic Regression: a Statistical View of Boosting” (1998)

gentleBoosting

- At each iteration:

  We chose $f_m(x)$ that minimizes the cost:

$$J(F + f_m) = \sum_{t=1}^{N} e^{-y_t(F(x_t)+f_m(x_t))}$$

  Instead of doing exact optimization, gentle Boosting minimizes the **approximation** of the error:

$$J(F) \propto \sum_{t=1}^{N} \boxed{e^{-y_t F(x_t)}} (y_t - f_m(x_t))^2$$

Weights at this iteration

At each iterations we just need to solve a weighted least squares problem

For more details: Friedman, Hastie, Tibshirani. "Additive Logistic Regression: a Statistical View of Boosting" (1998)

Weak Classifiers

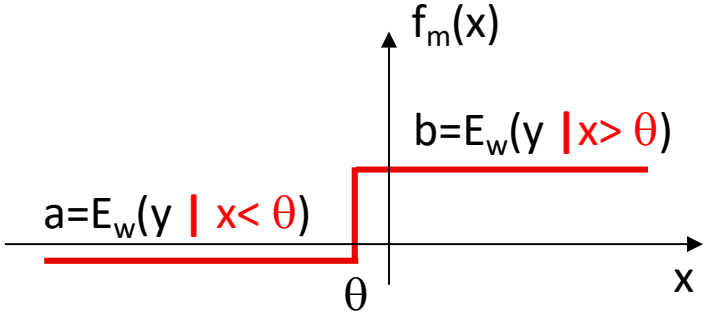
- 输入是一组加权的训练样本（x,y,w）。
- 回归树桩：简单但常用于物体检测。

- The input is a set of weighted training samples (x,y,w)

- Regression stumps: simple but commonly used in object detection.

$$f_m(x) = a[x_k < \theta] + b[x_k \geq \theta]$$

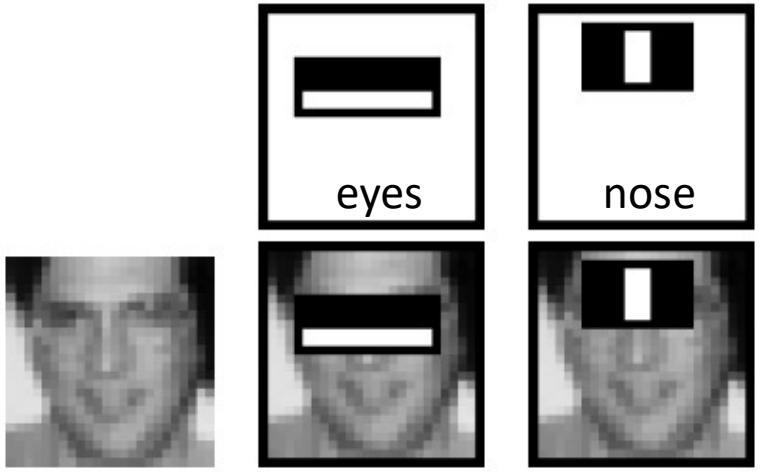
Four parameters: $\phi = [a, b, \theta, k]$

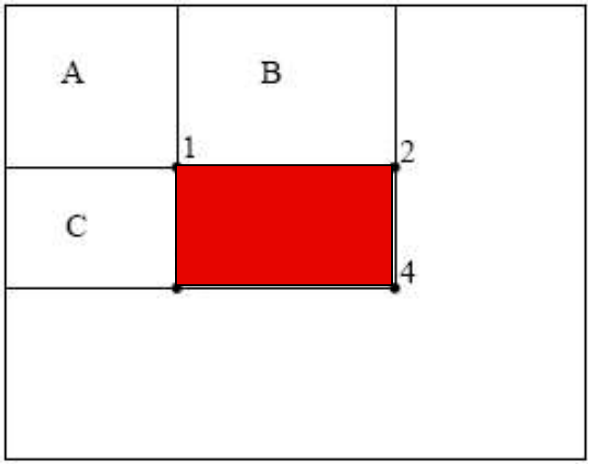


Features -> Weak Detectors

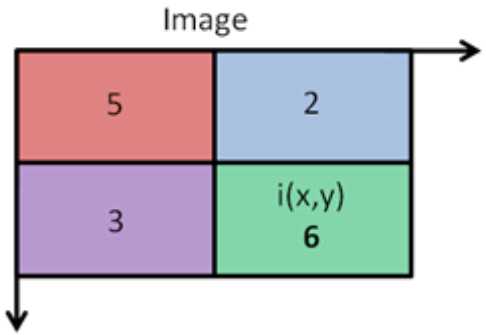
Haar filters and integral image

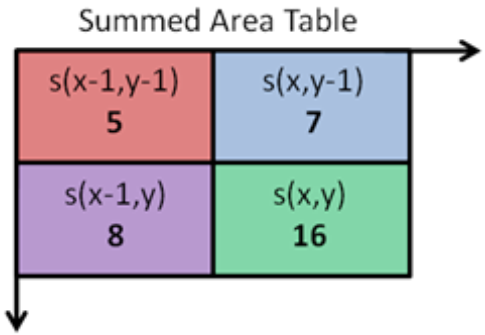
Viola and Jones, ICCV 2001

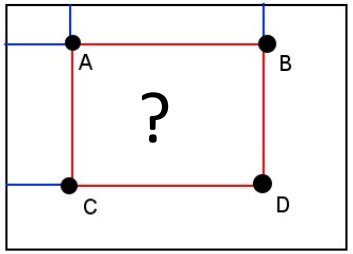




The average intensity in the block is computed with four sums independently of the block size.





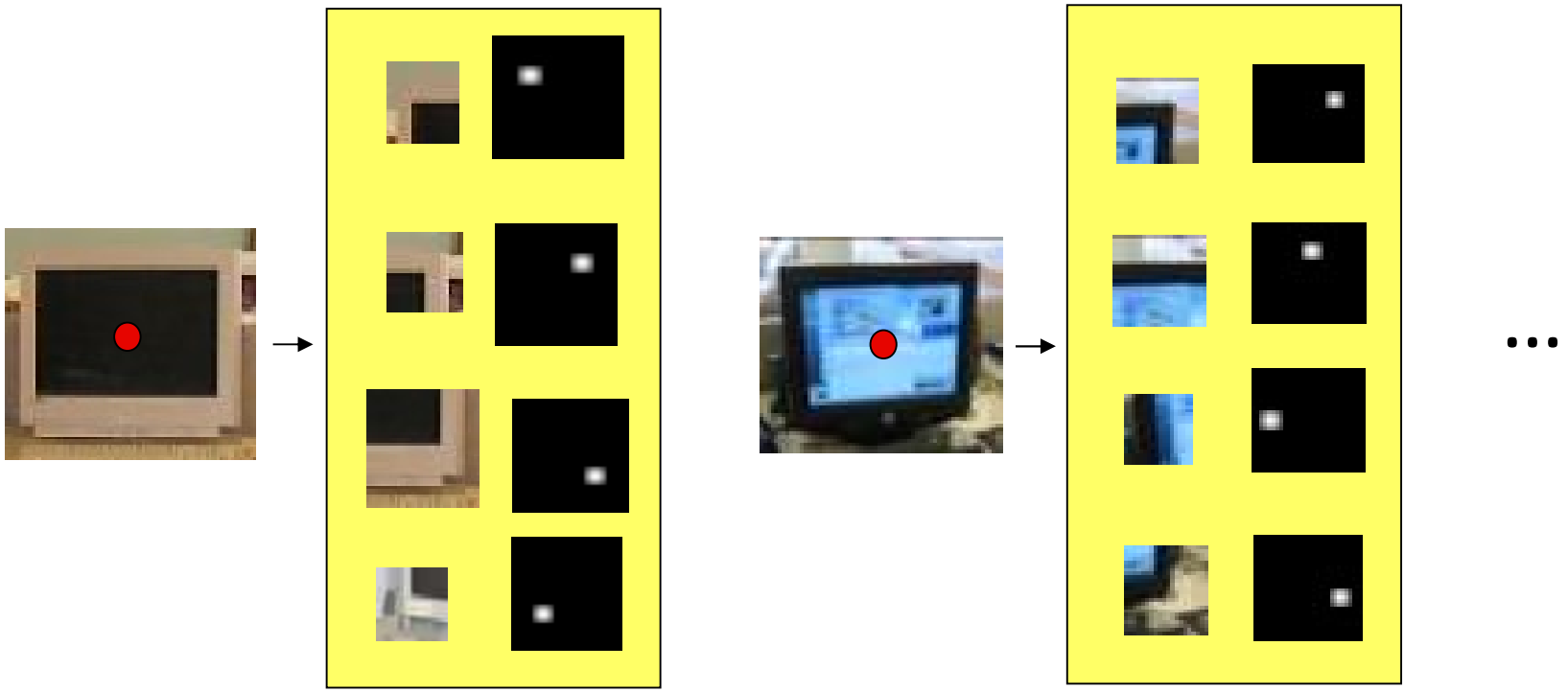


Sum = D - B - C + A

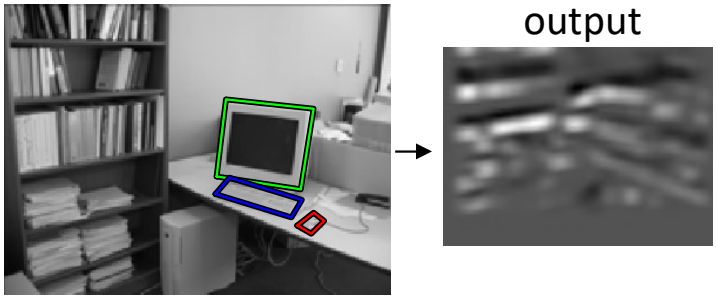
Features -> Weak Detectors

For screen detection, we may collect a set of part templates from a set of training objects to build feature set.

对于屏幕检测，我们可以从一组训练对象中收集一组零件模板来建立特征集。

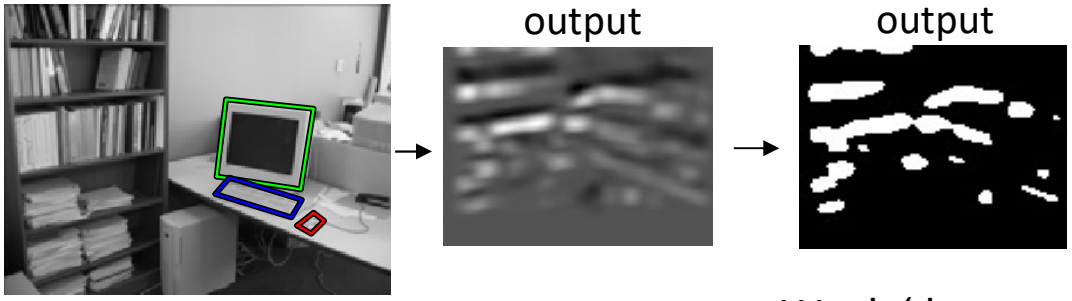


Example: Screen Detection



Feature output

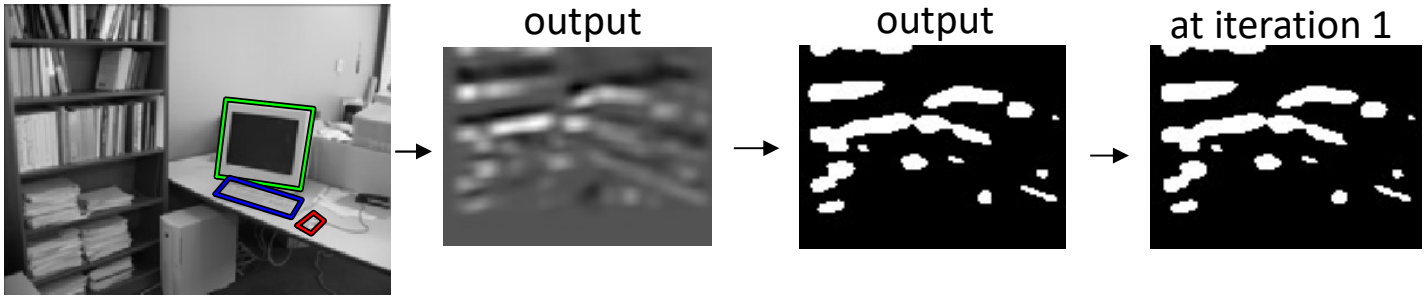
Example: Screen Detection



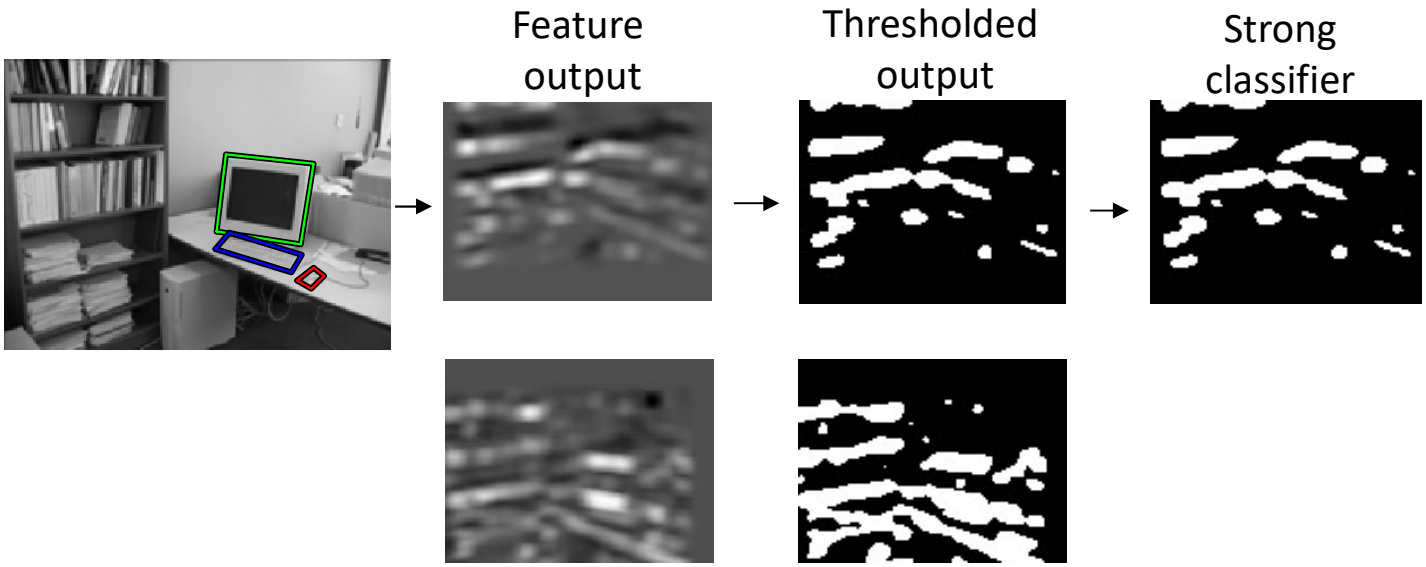
Weak ‘detector’

Produces many false alarms.

Example: Screen Detection



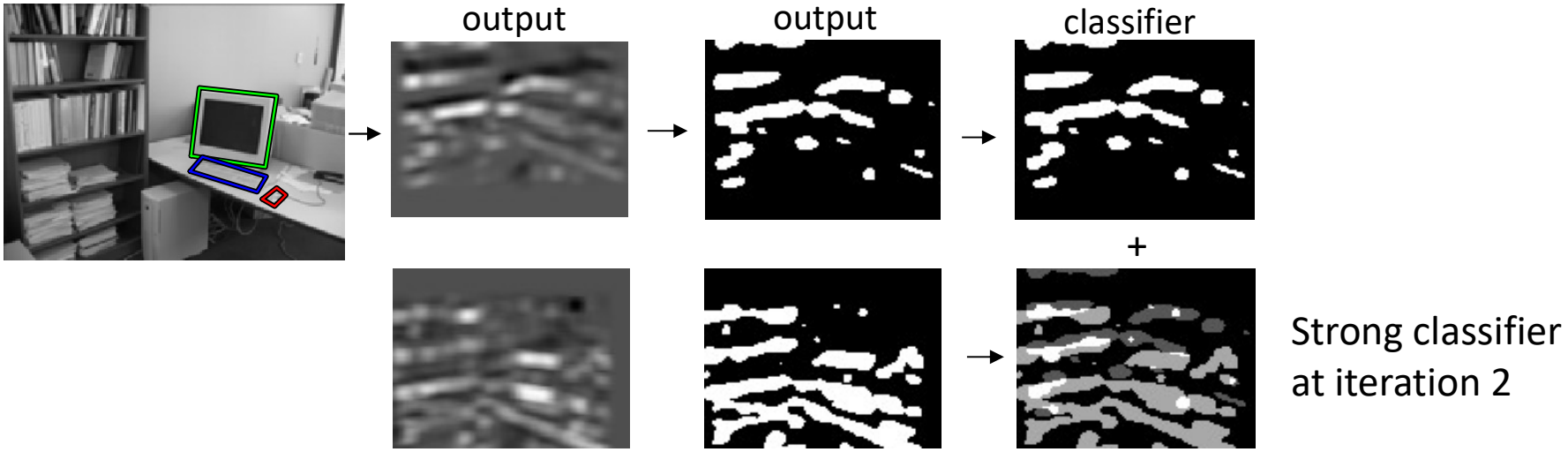
Example: Screen Detection



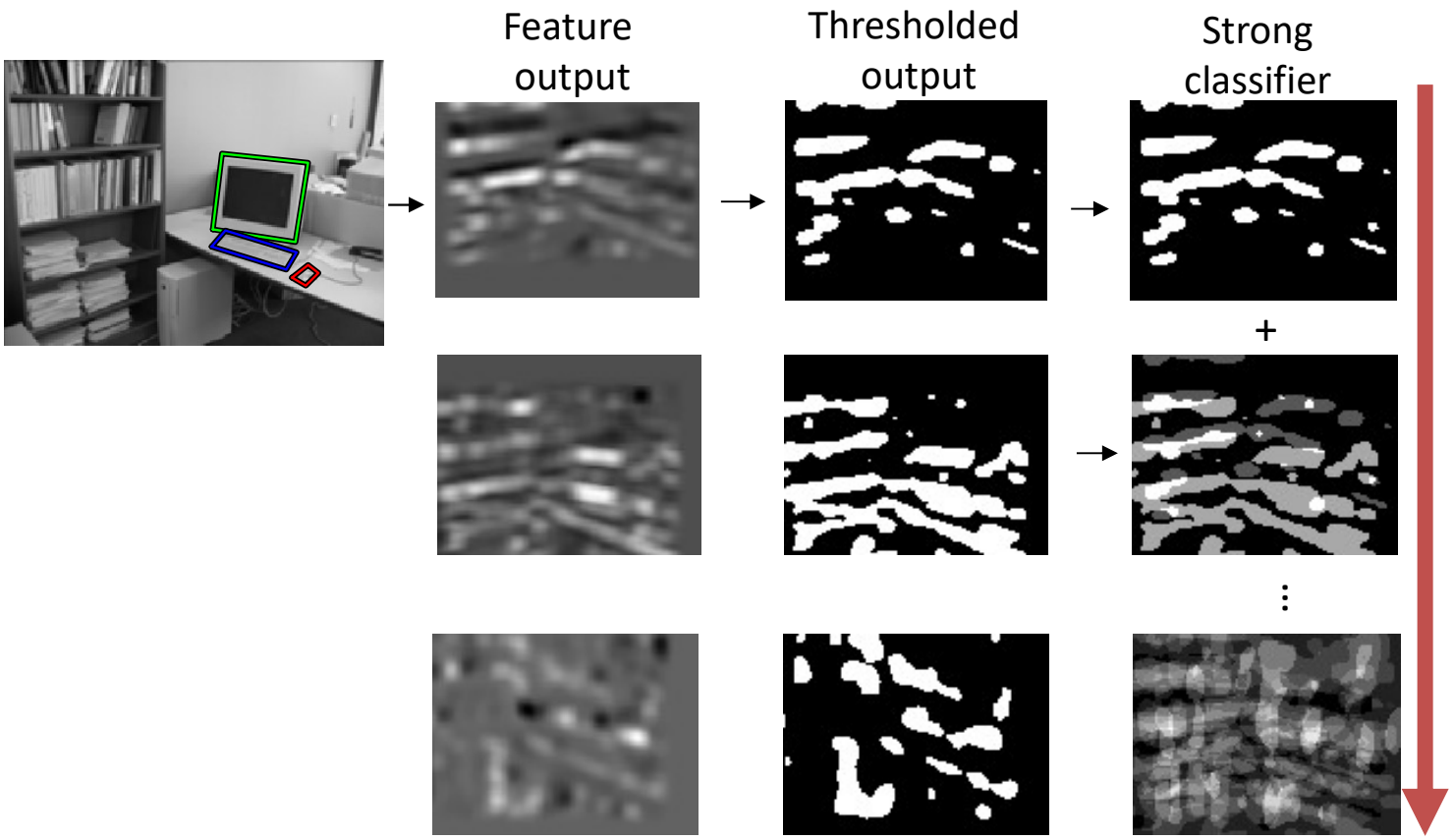
Second weak ‘detector’

Produces a different set of false alarms.

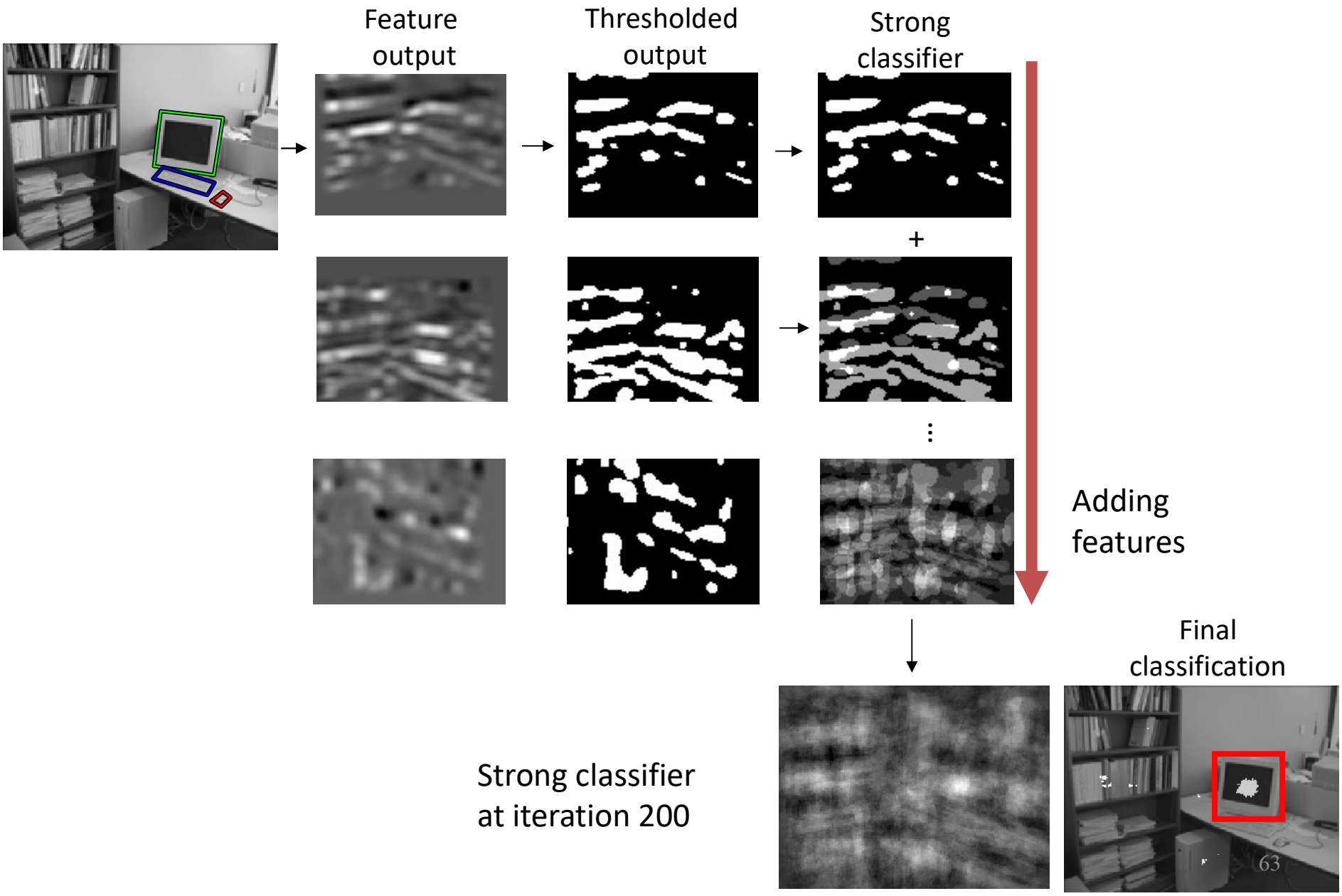
Example: Screen Detection



Example: Screen Detection

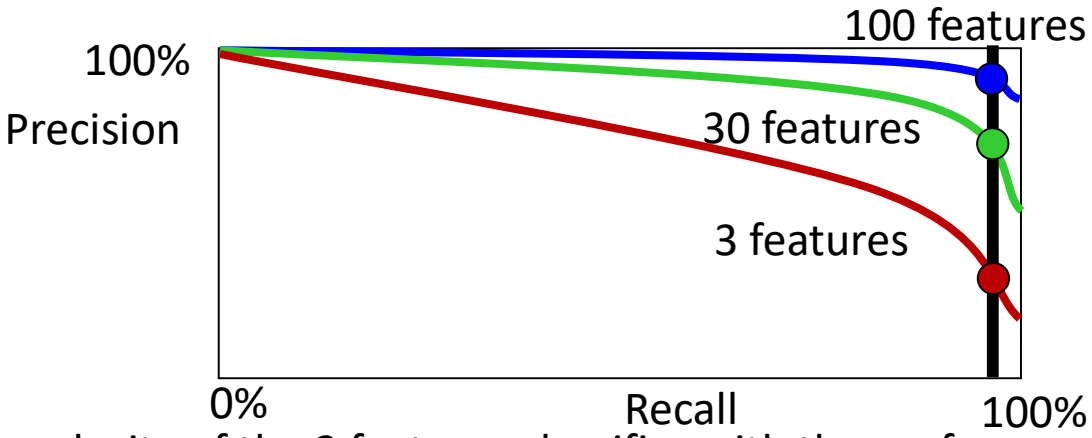


Example: Screen Detection

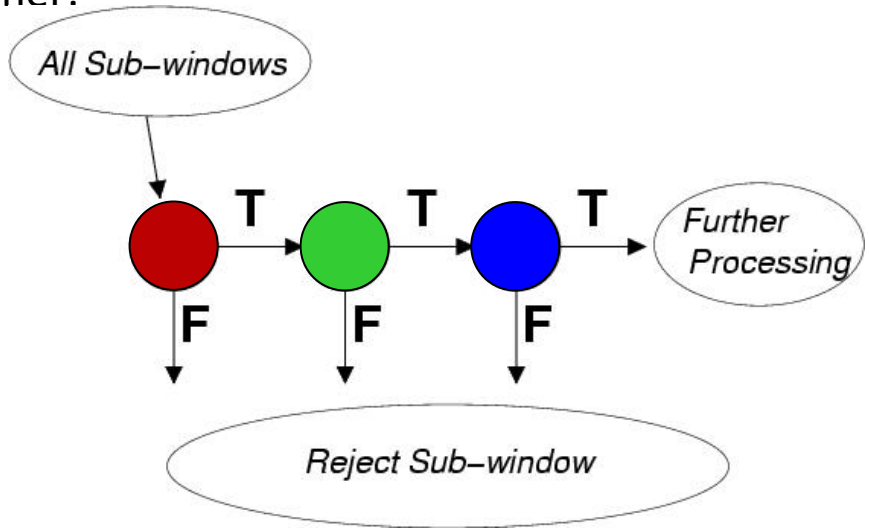


Cascade of classifiers

What is the motivation: some negative samples may be rejected based on few features!



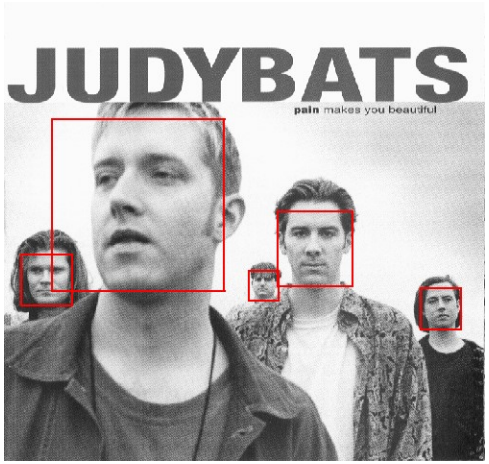
We want the complexity of the 3 features classifier with the performance of the 100 features classifier:

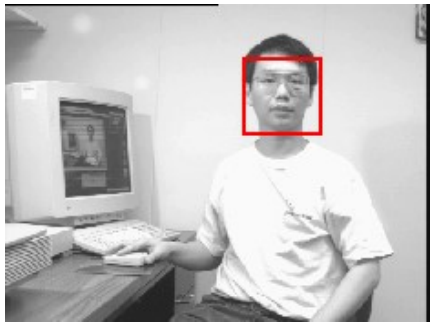


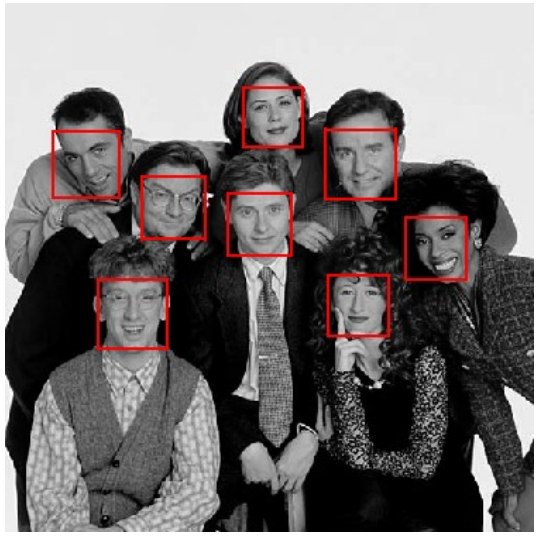
Select a threshold with high recall for each stage.

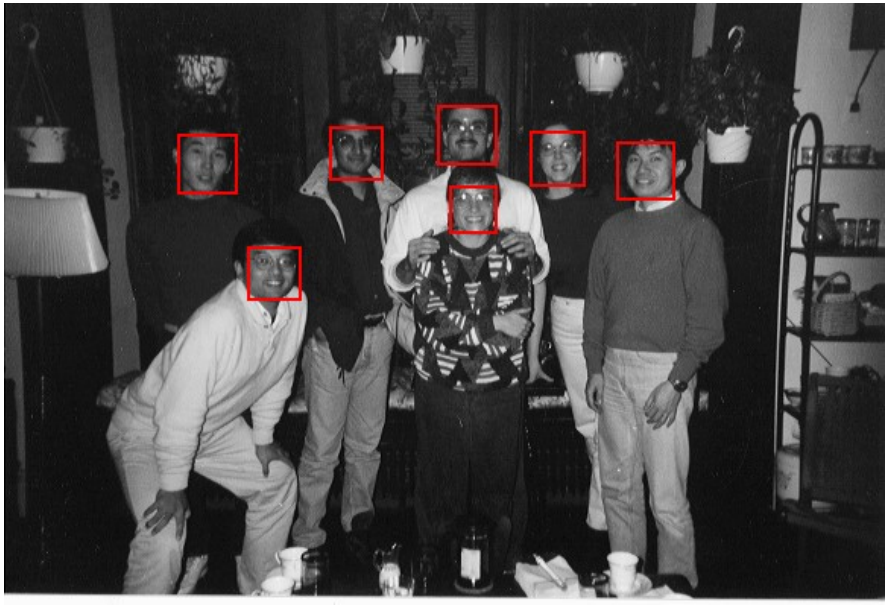
We increase precision using the cascade

Output of Face Detector on Test Images



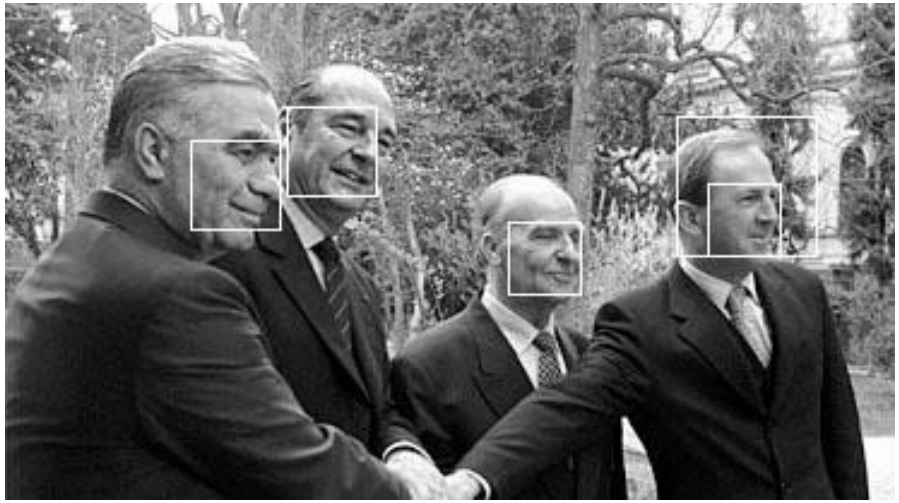






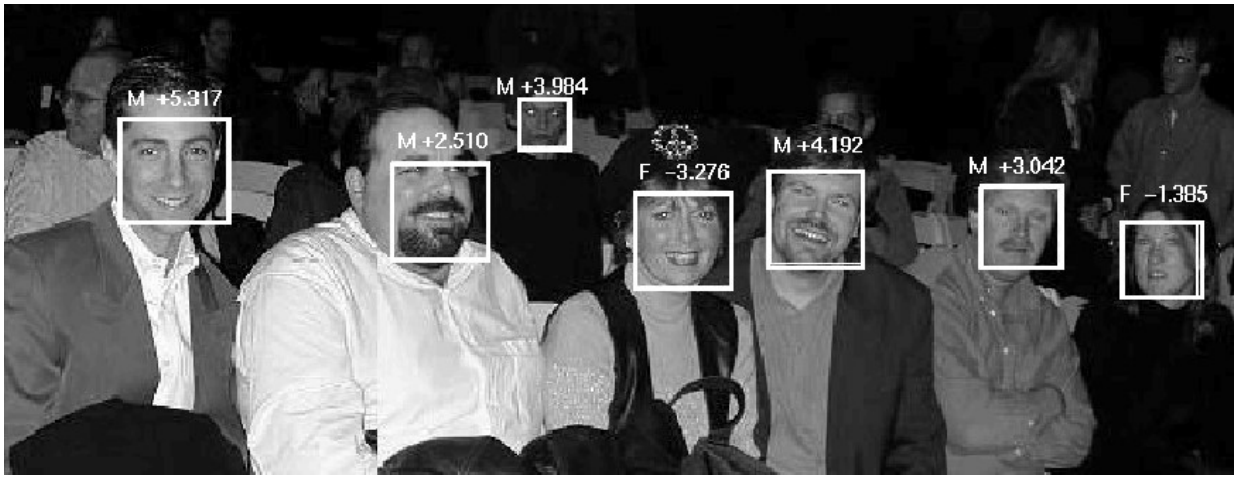
Other detection tasks

Facial Feature Localization



Profile Detection

Male vs. female



Profile Detection

“Head in the coffee beans problem”

Can you find the head in this image?

Weakness of Boosting

- 特征是在固定的位置提取的，因此不能变形（对可变形的物体不完美）
- 没有处理遮挡的机制
- 扩展到 "可变形模型 "+"和/或模型"？

- Features are extracted at fixed positions, and thus not deformable (not perfect for deformable objects)
- No mechanism for handling occlusion
- Extension to “deformable model” + “and/or model”?



Papers to Read and Study

- Friedman, Hastie, Tibshirani.
  Additive Logistic Regression: a Statistical View of Boosting (1998). Pdf

- Robert E. Schapire.
  The boosting approach to machine learning: An overview.
  In D. D. Denison, M. H. Hansen, C. Holmes, B. Mallick, B. Yu, editors,
  Nonlinear Estimation and Classification. Springer, 2003.
  Postscript or gzipped postscript.

- Ron Meir and Gunnar Rätsch.
  An introduction to boosting and leveraging.
  In *Advanced Lectures on Machine Learning (LNAI2600)*, 2003. Pdf.